TESIS – TE142599

SISTEM KEAMANAN DENGAN METODE ENKRIPSI *RSA* DAN *DIGITAL SIGNATURE ALGORITHM* UNTUK KOMUNIKASI BERGERAK PADA APLIKASI *INTELLIGENT TRANSPORTATION SYSTEM*

FARAH JIHAN AUFA
07111650030001

DOSEN PEMBIMBING
Dr. Ir. Endroyono, DEA.
Dr. Ir. Achmad Affandi, DEA.

PROGRAM MAGISTER
BIDANG KEAHLIAN TELEKOMUNIKASI MULTIMEDIA
DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI ELEKTRO
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2018

TESIS – TE142599

SISTEM KEAMANAN DENGAN METODE ENKRIPSI *RSA* DAN *DIGITAL SIGNATURE ALGORITHM* UNTUK KOMUNIKASI BERGERAK PADA APLIKASI *INTELLIGENT TRANSPORTATION SYSTEM*

FARAH JIHAN AUFA
07111650030001

DOSEN PEMBIMBING
Dr. Ir. Endroyono, DEA.
Dr. Ir. Achmad Affandi, DEA.

PROGRAM MAGISTER
BIDANG KEAHLIAN TELEKOMUNIKASI MULTIMEDIA
DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI ELEKTRO
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2018

LEMBAR PENGESAHAN

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Teknik (M.T)
di
Institut Teknologi Sepuluh Nopember

oleh:

Farah Jihan Aufa
NRP. 07111650030001

Tanggal Ujian : 05 Juli 2018
Periode Wisuda : September 2018

Disetujui oleh:

1. Dr. Ir. Endroyono, DEA. (Pembimbing)
   NIP: 196504041991021001
2. Dr. Ir. Achmad Affandi, DEA. (Pembimbing)
   NIP: 196510141990021001
3. Eko Setijadi, ST., MT., Ph.D. (Penguji)
   NIP: 197210012003121002
4. Dr. Istas Pratomo, ST., MT. (Penguji)
   NIP: 197903252003121001
5. Dr. Ir. Achmad Mauludiyanto, MT. (Penguji)
   NIP: 196109031989031001

Dekan Fakultas Teknologi Elektro

Dr. Tri Arief Sardjono, S.T., M.T.
NIP. 197002121995121001

Halaman ini sengaja dikosongkan

PERNYATAAN KEASLIAN TESIS

Dengan ini saya menyatakan bahwa isi keseluruhan Tesis saya dengan judul "**SISTEM KEAMANAN DENGAN METODE ENKRIPSI *RSA* DAN *DIGITSL SIGNATURE ALGORITHM* UNTUK KOMUNIKASI BERGERAK PADA APLIKASI *INTELLIGENT TRANSPORTATION SYSTEM***" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka. Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, 30 Mei 2018

Farah Jihan Aufa

NRP. 07111650030001

Halaman ini sengaja dikosongkan

SISTEM KEAMANAN DENGAN METODE ENKRIPSI *RSA* DAN *DIGITAL SIGNATURE ALGORITHM* UNTUK KOMUNIKASI BERGERAK PADA APLIKASI *INTELLIGENT TRANSPORTATION SYSTEM*

Nama mahasiswa : Farah Jihan Aufa
NRP : 07111650030001
Pembimbing : 1. Dr. Ir. Endroyono, DEA.
2. Dr. Ir. Achmad Affandi, DEA.

ABSTRAK

Intelligent Transportation System merupakan salah satu contoh perkembangan teknologi yang memadukan antara teknologi informasi dan telekomunikasi. Dalam penerapan *Intelligent Transportation System* ini dibutuhkan OBU (*On Board Unit*) yang diletakkan pada kendaraan. OBU berfungsi sebagai perangkat penghubung antara kendaraan dengan server melalui jaringan nirkabel. Pertukaran informasi yang dilakukan adalah informasi lalu lintas, informasi jalan, informasi darurat, jadwal transit, cuaca, pelayanan transportasi, dan lain sebagainya. Pada sistem komunikasi *Intelligent Transportation System*, pertukaran informasi sangatlah penting dan diharapkan kerahasiaannya. Sehingga dibutuhkan sistem keamanan yang dapat menjaga keutuhan, keaslian, dan kerahasiaan informasi tersebut.

Pada penelitian ini, penulis membuat simulasi dari sistem keamanan yang nantinya dapat diterapkan pada sistem komunikasi *Intelligent Transportation System* antara OBU hingga ke TMC (Traffic Management Center) menggunakan metode gabungan enkripsi RSA dan *Digital Signature Algorithm* dengan protokol UDP (User Datagram Protocol) sebagai jembatan untuk pengiriman informasi yang diolah. Metode ini dipilih karena tingkat kompleksitas yang relatif rendah yang menghasilkan ciphertext yang tidak terlalu banyak sehingga dapat menghemat bandwidth juga memiliki waktu komputasi yang rendah sehingga waktu delay pun juga rendah karena UDP mendukung pengiriman informasi yang relatif cepat dan hemat bandwidth. Hasil dari simulasi sistem yang dilakukan adalah didapatkan waktu komputasi yang relatif lebih lama pada sistem yang telah terintegrasi. Sistem

keamanan yang disimulasikan juga telah memenuhi persyaratan layanan keamanan dalam hal keutuhan data yang didapatkan pada komputer penerima (dalam hal ini TMC), komputer TMC dapat mendekripsi dan memverifikasi tanda tangan digital dengan benar sehingga data yang diterima sama dengan data yang dikirimkan.

Kata kunci: Digital Signature Algorithm, Enkripsi RSA, Intelligent Transportation System, User Datagram Protokol.

SECURITY SYSTEM WITH RSA ENCRYPTION METHOD AND DIGITAL SIGNATURE ALGORITHM FOR MOBILE COMMUNICATION IN INTELLIGENT TRANSPORTATION SYSTEM APPLICATION

By : Farah Jihan Aufa
Student Identity Number : 07111650030001
Supervisor(s) : 1. Dr. Ir. Endroyono, DEA.
2. Dr. Ir. Achmad Affandi, DEA.

ABSTRACT

Intelligent Transportation System is one example of technological development that combines information technology and telecommunication. In the application of Intelligent Transportation System is required OBU (On Board Unit) that placed on the vehicle. OBU functions as a connecting device between vehicle and server over wireless network. The exchange of information carried out is traffic information, road information, emergency information, transit schedules, weather, transportation services, etc. In the Intelligent Transportation System communication system, the exchange of information is very important and expected to be confidential. So it need a security system that can keep the integrity, authenticity, and confidentiality of such information. In this research, the author make simulation of security system which later can be applied to communication system of Intelligent Transportation System between OBU to TMC (Traffic Management Center) using combination method: RSA encryption and Digital Signature Algorithm with UDP (User Datagram Protocol) protocol as a bridge for delivery of processed information. This method is chosen because the relatively low level of complexity that produces little ciphertext that can save bandwidth and also has a low computing time so that the delay time is a low too because UDP supports the delivery of information is relatively fast and bandwidth-efficient. The result of the simulation of the system has relatively longer computation time than the system that has been integrated. The simulated security system also meets the requirements of security services in terms of the integrity of the data obtained on the receiving computer (in this case the TMC), the TMC computer can decrypt and verify the digital signature correctly so that the received data is the same as the transmitted data.

Key words: Digital Signature Algorithm, RSA Encryption, Intelligent Transportation System, User Datagram Protokol.

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Segala puji bagi Allah SWT yang telah memberikan bimbingan dan kelancaran sehingga penulis dapat menyelesaikan Tesis yang berjudul:

Sistem Keamanan dengan Metode Enkripsi RSA dan *Digital Signature Algorithm* untuk Komunikasi Bergerak pada aplikasi *Intelligent Transportation System*

Tesis ini disusun untuk memenuhi syarat menyelesaikan studi Strata-2 di jurusan Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember Surabaya. Selama pengerjaan, penulis mendapat banyak bantuan dari berbagai pihak. Oleh karena itu penulis ingin mengucapkan terima kasih kepada:

1. Orang tua dan mertua atas segala bantuannya baik moral, materiil, maupun doa.
2. Suami tercinta Bastian Ahmad Maladi, S.ST dan anak tersayang Muhammad Mehdy Mahdavikia atas segala motivasi, perhatian, dan doanya.
3. Bapak Dr. Ir. Endroyono, DEA dan Bapak Dr. Ir. Achmad Affandi, DEA selaku pembimbing Tesis yang telah banyak membantu, membimbing, memberi nasihat dan motivasi kepada penulis selama penulis belajar di ITS.
4. Dosen-dosen S-2 Telekomunikasi Multimedia, yang banyak memberikan ilmu, kritik, dan saran demi kelancaran penelitian Tesis.
5. Tim penelitian Intelligent Transportation System, yang telah membantu penulis selama penelitian dan penyusunan Tesis.
6. Teman-teman S-2 TMM, yang membantu penulis dalam memperkaya wawasan, khususnya yang mendukung dalam penyusunan Tesis.

Penulis menyadari bahwa Tesis ini masih jauh dari sempurna. Oleh karena itu, penulis mengharapkan kritik dan saran dari pembaca. Semoga Tesis ini dapat memberikan manfaat, baik bagi penulis maupun bagi pembacanya.

Surabaya, 30 Mei 2018

Farah Jihan Aufa

Halaman ini sengaja dikosongkan

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Kemacetan lalu lintas saat ini merupakan problem utama yang terjadi di kota-kota besar di Indonesia. Penyebab utama terjadinya kemacetan lalu lintas adalah karena tidak *demand* dan *supply* yaitu pertumbuhan jumlah kendaraan dengan kapasitas prasarana transportasi (jaringan jalan dan jaringan angkutan umum) yang ada [1]. *Intelligent Transportation System* merupakan salah satu contoh perkembangan teknologi yang memadukan antara teknologi informasi dan telekomunikasi. *Intelligent Transportation System* merupakan konsep kendaraan cerdas yang didukung oleh ICT yang dapat menurunkan tingkat kemacetan, meningkatkan kualitas layanan dan keselamatan lalu lintas, meningkatkan efisiensi transportasi, dan mengembangkan dan menerapkan inovasi teknologi baru dalam bidang transportasi termasuk pada sistem keamanannya. Dalam penerapannya, *Intelligent Transportation System* membutuhkan beberapa perangkat keras dan perangkat lunak yang mendukung jalannya sebuah *Intelligent Transportation System. OBU* (*On Board Unit*) adalah salah satu perangkat keras yang dibutuhkan untuk sistem komunikasi sebuah *Intelligent Transportation System* yang diletakkan pada kendaraan. OBU berfungsi sebagai perangkat penghubung antara kendaraan dengan server. Pertukaran informasi yang dilakukan dapat berupa informasi lalu lintas, informasi jalan, informasi darurat, jadwal transit, cuaca, pelayanan transportasi, dan lain sebagainya. Pada sistem komunikasi *Intelligent Transportation System*, pertukaran informasi sangatlah penting dan diharapkan kerahasiaannya. Sehingga dibutuhkan sistem keamanan yang dapat menjaga keutuhan, keaslian, dan kerahasiaan informasi tersebut. Selain itu, terdapat pula permasalahan pada sistem komunikasi bergerak diantaranya adalah delay yang tinggi, bandwidth kecil, tingkat keamanan rendah, dan sebagainya. Sehingga dibutuhkan sistem keamanan pada OBU dengan server yang dapat menyelesaikan permasalahan-permasalahan pada sistem komunikasi bergerak.

Beberapa penelitian sebelumnya meneliti tentang sistem keamanan menggunakan metode enkripsi RSA dan *Digital Signature Algorithm.* Novrizky Dwi Prasetya membuat dan menganalisa sistem pengamanan dokumen penawaran *E-procurement* menggunakan *Digital Signature Algorithm* 512 dan 1024 bit. Semakin panjang bit yang digunakan maka semakin lama pula waktu komputasinya [2]. Uma Somani, Kanika Lakhani, dan Manish Mundra menganalisa implementasi *Digital Signature* menggunakan algoritma enkripsi RSA untuk mendukung keamanan data pada *cloud computing.* Sehingga didapatkan sistem keamanan yang lebih baik dari metode RSA pada umumnya [3]. Rifki Sadikin menjelaskan pada buku yang berjudul kriptografi untuk keamanan jaringan bahwa *digital signature* dapat digunakan untuk mewujudkan layanan keamanan: otentikasi keaslian pesan, keutuhan pesan, dan *non-repudiation.* Sedangkan *enchipherment* dapat digunakan untuk mewujudkan layanan keamanan: otentikasi keutuhan pesan, kerahasiaan pesan, dan keutuhan pesan [4]. Ranbir Soram menjelaskan pada penelitiannya, bahwa faktorisasi integer dari RSA 1024 belum terpecahkan [5], sehingga RSA 1024 masih unggul dari RSA dengan bit-bit dibawahnya.

Pada penelitian ini, penulis membuat simulasi dari sistem keamanan yang nantinya dapat diterapkan pada sistem komunikasi *Intelligent Transportation System* antara OBU hingga ke TMC (Traffic Management Center) menggunakan metode gabungan enkripsi RSA dan *Digital Signature Algorithm* dengan protokol UDP (User Datagram Protocol) sebagai jembatan untuk pengiriman informasi yang diolah. RSA dipilih karena berdasarkan studi literatur dan referensi yang digunakan, RSA merupakan algoritma kriptografi asimetrik yang banyak digunakan karena kompleksitas dari operasi big integer RSA dapat meningkatkan tingkat keamanan. Pada penelitian ini digunakan RSA 1024 karena faktorisasi integer dari RSA 1024 masih belum terpecahkan hingga saat ini, sehingga RSA 1024 memiliki tingkat keamanan yang tinggi. Untuk menambah tingkat keamanannya, maka dipilih *Digital Signature Algorithm* sehingga pesan tidak hanya dienkripsi dan dekripsi saja, tetapi juga memiliki tanda tangan digital untuk menunjang otentikasi dan validasi data. *Digital signature Algorithm* yang digunakan pada penelitian ini adalah DSA 512. DSA 512 bit dipilih karena keunggulan pada waktu komputasinya yang relatif rendah dibandingkan dengan DSA 1024. Sehingga pada penelitian ini

dilakukan penggabungan antara RSA 1024 dan DSA 512. Gabungan kedua metode RSA 1024 dan DSA 512 menghasilkan ciphertext yang tidak terlalu banyak, sehingga dapat menghemat bandwidth, serta memiliki waktu komputasi yang rendah sehingga waktu delay pun juga rendah karena UDP mendukung pengiriman informasi yang relatif cepat dan hemat bandwidth. Dengan menerapkan sistem keamanan ini, komunikasi antara OBU dengan server akan aman dan terhindar dari serangan *attacker* atau biasa disebut MITM (*Man In The Middle*) karena metode RSA dan *Digital Signature Algorithm* ini dapat mewujudkan layanan keamanan berupa otentikasi keaslian pesan, keutuhan pesan, kerahasiaan pesan, dan *non-repudiation*.

Hasil dari simulasi sistem yang dilakukan adalah didapatkan waktu komputasi yang relatif lebih lama pada sistem yang telah terintegrasi. Pembangkitan kunci pada sistem terintegrasi memiliki waktu komputasi 3.88% lebih lama dari pembangkitan kunci sebelum terintegrasi. Proses enkripsi dan signing pada sistem terintegrasi memiliki waktu komputasi 10% lebih lama dari proses enkripsi dan signing sebelum terintegrasi. Proses dekripsi dan verifying pada sistem terintegrasi memiliki waktu komputasi 2.7% lebih lama dari proses dekripsi dan verifying sebelum terintegrasi. Sistem keamanan yang disimulasikan juga telah memenuhi persyaratan layanan keamanan dalam hal keutuhan data yang didapatkan pada komputer penerima (dalam hal ini TMC), komputer TMC dapat mendekripsi dan memverifikasi tanda tangan digital dengan benar sehingga data yang diterima sama dengan data yang dikirimkan.

1.2 Rumusan Masalah

1. Diperlukannya sistem keamanan dan otentikasi data pada sistem komunikasi *Intelligent Transportation System* khususnya pada pertukaran informasi OBU (*On Board Unit)* yang mempunyai kompleksitas rendah dan waktu komputasi yang cepat.
2. Algoritma yang digunakan untuk keperluan validasi dan autentikasi masih perlu dievaluasi.
3. Evaluasi kinerja gabungan antara metode enkripsi RSA dengan *Digital Signature Algorithm* pada sistem keamanan.

1.3 Tujuan

1. Membuat perangkat lunak untuk sistem keamanan data OBU (*On Board Unit*) pada komunikasi data *Intelligent Transportation System* menggunakan enkripsi RSA dan *Digital Sugnature Algorithm*
2. Mengimplementasikan aplikasi enkripsi dan otentikasi data pada OBU (*On Board Unit*) untuk mengamankan data OBU (*On Board Unit*) agar tidak dapat diserang oleh MITM
3. Melakukan evaluasi algoritma gabungan metode enkripsi RSA dan *Digital Signature Algorithm*

1.4 Batasan Masalah

1. Metode pengamanan data menggunakan enkripsi RSA
2. Metode otentikasi data menggunakan *Digital Signature Algorithm*
3. Sistem keamanan digunakan pada informasi berupa text
4. Simulasi data lokasi menggunakan pemrograman arduino
5. Menggunakan protokol UDP dalam pengiriman informasi
6. Menggunakan *Virtual Private Server* (VPS)
7. Sistem keamanan hanya ada pada layer aplikasi dan layer-layer lain diasumsikan aman

1.5 Kontribusi

Penelitian ini diharapkan dapat memberikan kontribusi berupa algoritma baru untuk keamanan data OBU pada sistem komunikasi *Intelligent Transportation System* menggunakan kombinasi metode enkripsi RSA dan *Digital Signature Algorithm*.

1.6 Sistematika Penulisan

Sistematika penulisan buku Tesis menjelaskan rincian dari laporan Tesis yang berisi tentang penjelasan mengenai tahapan-tahapan yang dilakukan dalam

penelitian Tesis. Berikut merupakan penjelasan dari tahapan-tahapan dari penulisan buku Tesis:

BAB 1 PENDAHULUAN

Bab ini berisi tentang latar belakang masalah, rumusan masalah, tujuan, batasan masalah, kontribusi penelitian, dan sistematika penulisan.

BAB 2 KAJIAN PUSTAKA

Bab ini menguraikan tentang landasan teori yang dibutuhkan untuk proses penelitian yang dilakukan seperti konsep dari *Intelligent Transportation System*, algoritma metode RSA, algoritma metode *Digital Signature Algorithm*, UDP (*User Datagram Protocol*), VPS (*Virtual Private Server*).

BAB 3 METODOLOGI PENELITIAN

Pada bagian ini akan dijelaskan secara rinci mengenai metodologi penelitian dalam merancang dan membuat program simulasi sistem keamanan pada *Intelligent Transportation System* khususnya pada pertukaran informasinya menggunakan algoritma enkripsi RSA dan *Digital Signature Algorithm*.

BAB 4 HASIL DAN PEMBAHASAN

Pada bab ini ditampilkan hasil pengujian simulasi sistem keamanan pada *Intelligent Transportation System*. Hasil pengujian yang ditampilkan dan dibahas merupakan hasil analisis terhadap parameter-parameter perbandingan yang ditentukan.

BAB 5 PENUTUP

Di bab ini, diberikan kesimpulan-kesimpulan berdasarkan hasil yang diperoleh. Selain itu, pada bab ini juga diberikan saran-saran untuk penelitian selanjutnya berkaitan dengan pengembangan dari penelitian yang dilakukan di dalam Tesis ini.

Halaman ini sengaja dikosongkan

BAB 2
KAJIAN PUSTAKA

2.1 Pendahuluan

Tinjauan pustaka yang diuraikan pada bab ini berisi konsep mengenai sistem keamanan dengan menggunakan RSA dan *Digital Signature Algorithm*. Bagian berikutnya yang dibahas dalam bab ini adalah semua literatur mengenai *Intelligent Transportation System,* sistem keamanan RSA dan *Digital Signature*, *user Datagram Protocol,* VPS (*Virtual Private Server*). Selain itu dalam bab ini juga diuraikan mengenai peralatan yang digunakan selama penelitian baik yang berupa *software* ataupun *hardware*.

2.2 Kajian Penelitian Terkait

Penelitian ini memiliki fokus pada penerapan sistem keamanan menggunakan metode enkripsi RSA dan *Digital Signature* Algorithm yang akan diterapkan pada OBU pada *Intelligent Transport System.* Beberapa penelitian yang mendukung tentang sistem keamanan telah dilakukan oleh beberapa peneliti sebelumnya.

Penelitian yang dilakukan oleh Novrizky Dwi Prasetya pada tahun 2014 membahas perancangan sistem keamanan pada *e-procurement* menggunakan algoritma *Digital Signature Algorithm* menggunakan 512 bit. Dari penelitian tersebut, didapatkan hasil bahwa semakin besar bit yang digunakan maka semakin lama pula waktu komputasi yang dibutuhkan pada proses pembangkitan kunci, proses *signing*, dan proses *verifiying* [2].

Penelitian selanjutnya yang dilakukan oleh Uma Somani, Kanika Lakhani, Manish Mundra adalah mengenai implementasi *Digital Signature* dengan enkripsi untuk mendukung keamanan data pada *cloud computing* [3].

Penelitian terkait tentang analisa performa metode enkripsi RSA telah dilakukan oleh Rabin Soram dan Engudam Sanahal Meitei pada tahun 2015. Penelitian yang dilakukan adalah analisa tentang performa dari algoritma RSA pada *virtual banking.* Dari penelitian tersebut, peneliti membandingkan waktu komputasi

yang dibutuhkan untuk pembangkitan kunci, enkripsi, dan dekripsi dengan mengunakan *key length* bit yang berbeda-beda mulai dari 1024 bit sampai 16384 bit dan didapatkan hasil yang optimal yaitu ketika menggunakan 3072 bit dengan waktu pembangkitan kunci 9141 ms, waktu enkripsi 3 ms, dan waktu dekripsi 567 ms [5]. Berikut adalah tabel yang menunjukkan deskripsi singkat dari penelitian sebelumnya.

Tabel 2.1 Dekripsi Penelitian Sebelumnya

No	Peneliti	Judul	Deskripsi
1	Novrizky Dwi Prasetya (2014)	Implementasi Sistem Pengamanan Dokumen Penawaran EProcurement Menggunakan Digital Signature Algorithm (DSA)	Penggunaan sistem komunikasi menggunakan algoritma DSA dengan *key length* 512 bit yang diterapkan pada *e-procurement.*
2	Uma Somani (2010)	Implementing Digital Signature with RSA Enryption Algorithm to Enhance the Data Security of Cloud in Cloud Computing	Implementasi keamanan data pada *cloud computing* dengan menggunakan algorithma enkripsi RSA yang diterapkan pada *digital signature*
3	Ranbir Soram, Engudam Sanahal Meitei	On the Performance of RSA in Virtual Banking	Menganalisa performa dari algortima RSA dengan mengubah *key length* bit

2.3 Teori Dasar

2.3.1 Intelligent Transportation System

Teknologi ITS (Intelligent Transport Sistem) adalah salah satu cabang artificial intelligence (AI) di bidang transportasi yang baru berkembang beberapa tahun terakhir untuk mengatasi kemacetan lalulintas di beberapa negara maju [1].

Lingkup ITS dapat berbeda pada masing-masing negara tergantung kepada kebijakan yang dibuat. Secara umum ITS mempunyai lingkup-lingkup sebagai berikut:

1. Advanced Traveller Information Sistem

Sistem ini secara prinsip adalah sistem informasi yang menjadi panduan kendaraan untuk mendapatkan routejalan yang optimal. Pada pengembangan selanjutnya sistem ini bahkan diharapkan mampu untuk membantu pengemudi mengontrol kendaraan agar sampai ditujuan dengan aman, nyaman dan lancar.

2. Advanced Traffic Management Sistem

Advanced Traffic Managent Sistemdigunakan oleh pengelola jalan untuk memantau lalulintas dan memberikan informasi realtime kepada pengguna jalan. Tujuan sistem ini agar lalulintas dapat dioptimalkan pada seluruh route alternatif yang ada, sehingga kemacetan dapat dihindari atau dikurangi dengan memberikan saran kepada pemakai jalan. Sistem ini jugamemberikan informasi adanya hambatan atau kecelakaan pada routeyang akan ditempuh, sehingga pengemudi dapat memakai alternatif routelain.

3. Incident Management Sistem

Incident Management Sistem adalah sistem informasi yang digunakan untuk berbagai kejadian darurat, misalkan kecelakaan, longsor atau bencana lainnya. Berdasarkan hasil pemantauan sensor-sensor pada Traffic Management Sistem, pengelola jalan atau pihak yang berwenang dapat memperoleh informasi lebih awal. Informasi dapat berupa besarnya kecelakaan, fatalitas kecelakaan, jumlah ambulans yang diperlukan, tenaga medis yang harus dikirim, alat penolong yang harus didatangkan dan sebagainya.

4. Electronic Toll Collection Sistem

Persoalan klasik pada jalan tol adalah lama waktu yang diperlukan untuk transaksi pelanggan di gerbang tol. Electronic Toll Collection diterapkan untuk mempersingkat waktu transaksi di gerbang tol.

5. Assistance For Safe Driving

Assistance for Safe Driving adalah bentuk dari ITS yang sangat maju. Kendaraan dilengkapi dengan sejumlah sensor yang dapat mengarahkan

pengemudi unuk berkendara dengan aman. Sensor tersebut dihubungkan dengan sebuah komputer yang terpasang pada kendaraan.

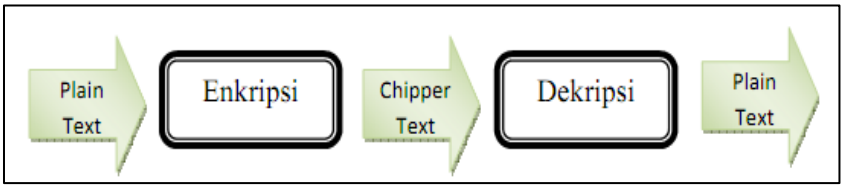
6. Support for Public Transportation

ITS jenis ini diterapkan pada moda transpotasi umum, misalnya: pesawat terbang, bus, kapal laut, ferri, monoraildan kereta api. Selain diterapkan pada wahana transportasi publik, sistem ini juga diterapkan pada prasarana transportasi publik seperti: stasiun kereta api, terminal bus, shelterbus, pelabuhan dan bandara [6].

2.3.2 Kriptografi

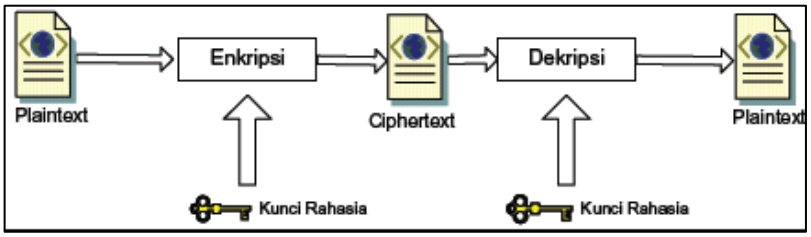
Kriptografi mempunyai 2 (dua) bagian yang penting yaitu enkripsi dan dekripsi. Enkripsi adalah proses dari penyandian pesan asli menjadi pesan yang tidak dapat diartikan seperti aslinya. Sedangkan deskripsi sendiri berarti merubah pesan yang sudah disandikan menjadi pesan aslinya. Pesan asli disebut *plaintext*, sedangkan pesan yang sudah disandikan disebut *chiphertext*.

Pada Gambar 2.1 dapat dilihat bahwa masukan berupa *plaintext* kemudian akan masuk ke dalam blok enkripsi dan keluarannya berupa *ciphertext*, kemudian *ciphertext* akan masuk ke dalam blok deskripsi dan keluarannya akan menjadi *plaintext* semula.



Gambar 2.1 Proses Enkripsi dan Dekripsi

Ada 2 (dua) model algoritma enkripsi yang menggunakan kunci, yaitu kunci simetrik dan kunci asimetrik. Enkripsi kunci simetrik yang biasanya disebut enkripsi konvensional adalah enkripsi yang menggunakan kunci yang sama untuk enkripsi maupun dekripsi, dari Gambar 2.2 terlihat bahwa untuk mengenkripsi maupun mendekripsi pesan hanya menggunakan satu buah kunci (K) saja.



Gambar 2.2 Enkripsi – Dekripsi Kunci Simetrik

Penggunaan metode ini membutuhkan persetujuan antara pengirim dan penerima tentang kunci sebelum mereka saling mengirim pesan. Keamanan dari kunci simetrik tergantung pada kerahasiaan kunci, apabila seorang penyusup dapat menemukan kunci maka dengan mudah dapat membaca pesan yang sudah dienkripsi. Enkripsi kunci simetrik dapat dibagi kedalam 2 (dua) kelompok yaitu metode *stream cipher* dan metode *block c*ipher.

Stream cipher yaitu proses mengenkripsi aliran data informasi (*stream*) bit per bitnya menjadi data acak (*cipher*) secara kontinyu. Sedangkan Block cipher merupakan proses mengenkripsi data informasi per blok data menjadi data acak (*Cipher*).

Enkripsi kunci asimetrik (biasa disebut enkripsi kunci publik) dibuat sedemikian rupa sehingga kunci yang dipakai untuk enkripsi berbeda dengan kunci yang dipakai untuk dekripsi. Enkripsi kunci publik disebut demikian karena kunci untuk enkripsi boleh disebarluaskan kepada umum sedangkan kunci untuk mendekripsi hanya disimpan oleh orang yang bersangkutan. Enkripsi asimetrik dapat ditulis seperti berikut:

$$Ek\ (P) = C \tag{2.1}$$

$$Dk\ (C) = P \tag{2.2}$$

Dimana :

Ek = Enkripsi

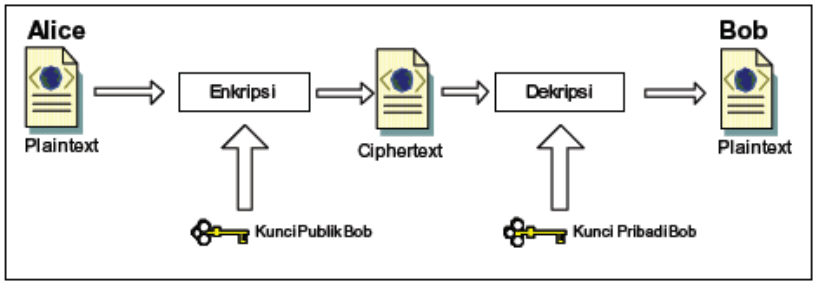
Dk = Dekripsi

P = Plaintext

C = Ciphertext

Contohnya seperti pada Gambar 2.3 bila seseorang ingin mengirim pesan kepada orang lain maka orang tersebut menggunakan kunci publik orang tersebut

untuk mengenkripsi pesan yang kita kirim kepadanya lalu orang tersebut akan mendekripsi pesan tersebut dengan kunci privat miliknya.



Gambar 2.3 Enkripsi Kunci Asimetrik

2.3.3 Algoritma RSA

Dari sekian banyak algoritma kriptografi kunci-publik yang pernah dibuat, algoritma yang paling populer adalah algoritma RSA. Algoritma RSA dibuat oleh 3 orang peneliti dari MIT (*Massachussets Institute of Technology*) pada tahun 1976, yaitu: Ron (R)ivest, Adi (S)hamir, dan Leonard (A)dleman.

Keamanan algoritma RSA terletak pada sulitnya memfaktorkan bilangan yang besar menjadi faktor-faktor prima. Pemfaktoran dilakukan untuk memperoleh kunci pribadi. Selama pemfaktoran bilangan besar menjadi faktor-faktor prima belum ditemukan algoritma yang mangkus, maka selama itu pula keamanan algoritma RSA tetap terjamin [7].

Besaran-besaran yang digunakan pada algoritma RSA:

1. p dan q bilangan prima (rahasia)
2. $r = p \cdot q$ (tidak rahasia)
3. $\phi(r) = (p-1)(q-1)$ (rahasia)
4. PK (kunci enkripsi) (tidak rahasia)
5. SK (kunci dekripsi) (rahasia)
6. X (plainteks) (rahasia)
7. Y (cipherteks) (tidak rahasia)

Algoritma RSA didasarkan pada teorema Euler (lihat bahan kuliah Teori Bilangan Bulat) yang menyatakan bahwa:

$$a^{\phi(r)} \equiv 1 \pmod{r} \qquad (2.3)$$

yang dalam hal ini,

1. a harus relatif prima terhadap r
2. $\phi(r) = r(1 - 1/p_1)(1 - 1/p_2) \ldots (1 - 1/p_n)$, yang dalam hal ini $p_1, p_2, \ldots, p_n$ adalah faktor prima dari r.

$\phi(r)$ adalah fungsi yang menentukan berapa banyak dari bilangan-bilangan 1, 2, 3, …, r yang relatif prima terhadap r. Berdasarkan sifat $a^m \equiv b^m \pmod{r}$ untuk m bilangan bulat ≥ 1, maka persamaan 2.3 dapat ditulis menjadi

$$a^{m\phi(r)} \equiv 1^m \pmod{r} \tag{2.4}$$

atau

$$a^{m\phi(r)} \equiv 1 \pmod{r} \tag{2.5}$$

Bila a diganti dengan X, maka persamaan 2.4 menjadi

$$X^{m\phi(r)} \equiv 1 \pmod{r} \tag{2.6}$$

Berdasarkan sifat $ac \equiv bc \pmod{r}$, maka bila persamaan 2.6 dikali dengan X menjadi:

$$X^{m\phi(r)+1} \equiv X \pmod{r} \tag{2.7}$$

yang dalam hal ini X relatif prima terhadap r. Misalkan SK dan PK dipilih sedemikian sehingga

$$SK \cdot PK \equiv 1 \pmod{\phi(r)} \tag{2.8}$$

atau

$$SK \cdot PK = m\phi(r) + 1 \tag{2.9}$$

Sulihkan 2.8 ke dalam persamaan 2.6 menjadi:

$$X^{SK \cdot PK} \equiv X \pmod{r} \tag{2.10}$$

Persamaan 2.10 dapat ditulis kembali menjadi

$$(X^{PK})^{SK} \equiv X \pmod{r} \tag{2.11}$$

Yang artinya, perpangkatan X dengan PK diikuti dengan perpangkatan dengan SK menghasilkan kembali X semula. Berdasarkan persamaan 2.11, maka enkripsi dan dekripsi dirumuskan sebagai berikut:

$$E_{PK}(X) = Y \equiv X^{PK} \bmod r \tag{2.12}$$

$$D_{SK}(Y) = X \equiv Y^{SK} \bmod r \tag{2.13}$$

Karena $SK \cdot PK = PK \cdot SK$, maka enkripsi diikuti dengan dekripsi ekivalen dengan dekripsi diikuti enkripsi:

$$E_{SK}(D_{SK}(X)) = D_{SK}(E_{PK}(X)) \equiv X^{PK} \bmod r \tag{2.14}$$

Oleh karena $X^{PK} \bmod r \equiv (X + mr)^{PK} \bmod r$ untuk sembarang bilangan bulat m, maka tiap plainteks X, $X + r$, $X + 2r$, …, menghasilkan cipherteks yang sama. Dengan kata lain, transformasinya dari banyak ke satu. Agar transformasinya satu-ke-satu, maka X harus dibatasi dalam himpunan $\{0, 1, 2, \ldots, r - 1\}$ sehingga enkripsi dan dekripsi tetap benar seperti pada persamaan 2.12 dan 2.13.

2.3.3.1 Prosedur Pembuatan Kunci

Setelah mengetahui parameter-parameter yang digunakn dalam metode RSA, maka dapat dilanjutkan dengan prosedur pembangkitan kunci yaitu sebagai berikut:

1. Pilih dua buah bilangan prima sembarang, p dan q.
2. Hitung $r = p \cdot q$. Sebaiknya $p \neq q$, sebab jika $p = q$ maka $r = p^2$ sehingga p dapat diperoleh dengan menarik akar pangkat dua dari r.
3. Hitung $\phi(r) = (p - 1)(q - 1)$.
4. Pilih kunci publik, PK, yang relatif prima terhadap $\phi(r)$.
5. Bangkitkan kunci rahasia dengan menggunakan persamaan (5), yaitu $SK \cdot PK \equiv 1 \pmod{\phi(r)}$.

Perhatikan bahwa $SK \cdot PK \equiv 1 \pmod{\phi(r)}$ ekivalen dengan $SK \cdot PK = 1 + m\phi(r)$, sehingga SK dapat dihitung dengan:

$$SK = \frac{1 + m\phi(r)}{PK} \tag{2.15}$$

Akan terdapat bilangan bulat *m* yang menyebabkan memberikan bilangan bulat *SK*.

Catatan: *PK* dan *SK* dapat dipertukarkan urutan pembangkitannya. Jika langkah 4 diganti dengan "Pilih kunci rahasia, SK, yang …", maka pada langkah 5 kita menghitung kunci publik dengan rumus yang sama.

2.3.3.2 Enkripsi

Plainteks disusun menjadi blok-blok $x_1, x_2, \ldots$, sedemikian sehingga setiap blok merepresentasikan nilai di dalam rentang 0 sampai $r - 1$.Setiap blok x_i dienkripsi menjadi blok y_i dengan rumus

$$y_i = x_i^{PK} \bmod r \tag{2.16}$$

2.3.3.3 Dekripsi

Setiap blok cipherteks y_i didekripsi kembali menjadi blok x_i dengan rumus

$$x_i = y_i^{SK} \bmod r \tag{2.17}$$

2.3.3.4 Kekuatan dan Keamanan RSA

Keamanan algoritma *RSA* terletak pada tingkat kesulitan dalam memfaktorkan bilangan non prima menjadi faktor primanya, yang dalam hal ini $r = p \times q$. Sekali r berhasil difaktorkan menjadi p dan q, maka $\phi(r) = (p - 1)(q - 1)$ dapat dihitung. Selanjutnya, karena kunci enkrispi *PK* diumumkan (tidak rahasia), maka kunci dekripsi *SK* dapat dihitung dari persamaan $PK \cdot SK \equiv 1 \pmod{\phi(r)}$.

Penemu algoritma *RSA* menyarankan nilai p dan q panjangnya lebih dari 100 digit. Dengan demikian hasil kali $r = p \times q$ akan berukuran lebih dari 200 digit. Menurut Rivest dan kawan-kawan, uasaha untuk mencari faktor bilangan 200 digit membutuhkan waktu komputasi selama 4 milyar tahun. (dengan asumsi bahwa

algoritma pemfaktoran yang digunakan adalah algoritma yang tercepat saat ini dan komputer yang dipakai mempunyai kecepatan 1 milidetik).

2.3.4 Digital Signature Algorithm

Pada bulan Agustus 1991, NIST (*The National Institute of Standard and Technology*) mengumumkan algoritma sidik digital yang disebut *Digital Signature Algorithm* (DSA). DSA dijadikan sebagai bakuan (*standard*) dari *Digital Signature Standard* (DSS). DSS adalah standard, sedangkan DSA adalah algoritma. Standard tersebut menggunakan algoritma ini, sedangkan algoritma adalah bagian dari standard (selain DSA, DSS menggunakan *Secure Hash Algorithm* atau SHA sebagai fungsi *hash*) DSA termasuk ke dalam sistem kriptografi kunci-publik. Meskipun demikian, DSA tidak dapat digunakan untuk enkripsi. Digital Signature Algorithm (DSA) merupakan varian dari skema tanda tangan ElGamal. Perbedaan mendasar dengan tanda tangan ElGamal adalah jumlah perhitungan eksponensial pada proses verifikasi dikurangi dari tiga menjadi dua. Perbedaan yang terpenting adalah banyaknya digit eksponen dibatasi 160 bit, tidak seperti tanda tangan ElGamal yang tidak dibatasi, tergantung pada pemilihan bilangan prima p, paling tidak sebesar 768 bit. DSA menggunakan SHA (Secure Hash Algorithm) sebagai fungsi hash. DSA mempunyai dua fungsi utama:

1. Pembentukan tanda tangan digital (*signature generation*)
2. Pemeriksaan keabsahan tanda tangan digital (*signature verification*).

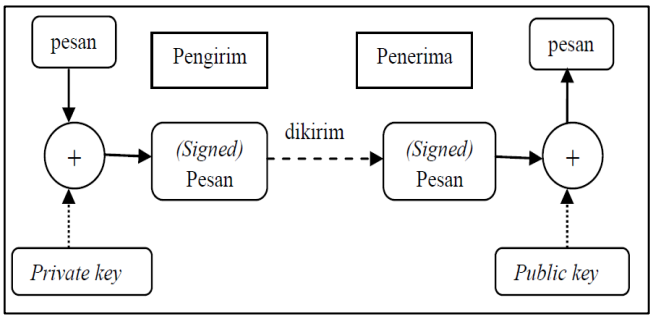
Tanda tangan digital DSA berbentuk sepasang besar angka yang ditampilkan computer sebagai string dari digit biner. Tanda tangan digital dihitung dengan menggunakan sejumlah aturan dan sejumlah parameter sehingga identitas pemilik dan integritas data dapat diverifikasi. Pembuat tanda tangan menggunakan kunci privat untuk membuat tanda tangan; sedangkan kunci publik, yang berkorespodensi dengan kunci privat namun tidak sama, digunakan untuk memverifikasi tanda tangan. Setiap user memiliki sepasang kunci publik dan kunci privat. Kunci publik diasumsikan diketahui public secara umum, sedangkan kunci privat tidak pernah disebar [8].

Sebagaimana halnya pada algoritma kriptografi kunci-publik, DSA menggunakan dua buah kunci, yaitu kunci publik dan kunci rahasia. Pembentukan

tanda tangan digital. menggunakan kunci rahasia pengirim, sedangkan verifikasi tanda tangan digital menggunakan kunci publik pengirim. Berikut ini akan dipaparkan mengenai cara kerja DSA.

Skema digital signature terdiri dari tiga algoritma:

1. Algoritma untuk membangkitkan *private key* dan *publik key*-nya.
2. Algoritma untuk memberi digital signature pada dokumen jika disediakan dokumen dan *private key*.
3. Algoritma untuk verifikasi tanda tangan digital jika disediakan dokumen, *public key*, dan digital signature



Gambar 2.4 Blok Diagram Digital Signature

DSA mempunyai properti berupa parameter sebagai berikut:

1. p, adalah bilangan prima dengan panjang L bit, yang dalam hal ini 512 ≤ L ≤ 1024 dan L harus kelipatan 64. Parameter p bersifat publik.
2. q, bilangan prima 160 bit, merupakan factor dari p – 1. Dengan kata lain, (p – 1) mod q = 0. Parameter q bersifat publik.
3. $g = h^{(p-1)/q} \bmod p$, yang dalam hal ini h < p – 1 sedemikian sehingga $h^{(p-1)/q} \bmod p > 1$. Parameter g bersifat publik.
4. x, adalah bilangan bulat kurang dari q. Parameter x adalah kunci privat.
5. $y = g^x \bmod p$, adalah kunci publik.
6. m, pesan yang akan diberi tanda-tangan

2.3.4.1 Prosedur Pembuatan Kunci

1. Pilih bilangan prima p dan q, yang dalam hal ini (p-1)mod q = 0
2. Hitung $g = h^{(p-1)/q} \bmod p$, yang dalam hal ini 1<h< p -1dan $h^{(p-1)/q} \bmod p>1$
3. Tentukan kunci privat x, yang dalam hal ini x< q.

4. Hitung kunci publik $y = g^x \bmod p$.

2.3.4.2 Proses Pemberian Tanda Tangan

1. Bangkitkan bilangan k secara acak untuk setiap pesan, di mana $0 < k < q$.
2. Hitung $r = (g^k \bmod p) \bmod q$
3. Hitung $s = (k^{-1}(\text{SHA-1}(m) + x*r)) \bmod q$, di mana SHA-1(m) merupakan fungsi hash SHA-yang diterapkan terhadap pesan m.
4. Tanda tangan digitalnya adalah (r, s)

2.3.4.3 Proses Verifikasi Tanda Tangan

1. Hitung $w = (s)^{-1} \bmod q$
2. Hitung $u1 = (\text{SHA-1}(m)*w) \bmod q$
3. Hitung $u2 = (r*w) \bmod q$
4. Hitung $v = ((g^{u1}*y^{u2}) \bmod p) \bmod q$
5. Tanda tangan digital dianggap valid jika memenuhi v = r

2.3.5 Layanan Keamanan Jaringan

Lembaga internasional yang bernama **Internasional Telecommunication Union – Telecommunication Standardiation Sector (ITU-T)** mendefinisikan beberapa jenis layanan *(services)* dan mekanisme *(mechanism)* keamanan jaringan. Layanan-layanan keamanan jaringan didefinisikan berdasarkan kebutuhan yang harus disediakan untuk memenuhi permintaan terhadap keamanan jaringan. Pada bagian ini akan dibahas terlebih dahulu jenis-jenis layanan keamanan jaringan berdasarkan rekomendasi ITU-T pada dokumen X.800, (ITU,1991).

1. Otentikasi

   Ketika *Alice* melakukan komunikasi data dengan *Bob* melalui jaringan data ada dua persoalan yang muncul, yaitu bagaimana *Alice* bisa yakin bahwa ia berkomunikasi dengan *Bob* dan bagaimana *Alice* bisa yakin bahwa data yang diterimanya memang berasal dari *Bob.* Layanan otentikasi memastikan keduanya. Layanan pertama disebut dengan otentikasi entitas yaitu layanan keamanan jaringan yang memberikan kepastian terhadap identitas subuah

entitas yang terlibat dalam komunikasi data. Sedangkan layanan kedua disebut dengan otentikasi keaslian data yaitu layanan yang memberikan kepastian terhadap sumber sebuah data.

2. Kendali Akses

   Kendali akses adalah layanan keamanan jaringan yang menghalangi penggunaan tidak terotoritasasi terhadap sumber daya. Pada aplikasi jaringan biasanya kebijakan kemampuan ditentukan oleh jenis pengguna. Misalnya sebuah data rekam medic elektronik hanya dapat diakses oleh pasien dan paramedis yang terlibat.

3. Kerahasiaan Data

   Kerahasiaan data adalah layanan keamanan jaringan yang memproteksi data tertransmisi terhadap pengungkapan oleh pihak yang tidak berwenang. Misalnya *Alice* mengirim data rahasia melalui internet ke *Bob* pada saat yang sama *Eve* mampu membaca data rahasia yang terkirim itu melalui *router* maka layanan kerahasiaan data memastikan bahwa data rahasia meskipun mampu dibaca oleh *Eve* tetap bersifat rahasia.

4. Keutuhan Data

   Keutuhan data adalah layanan keamanan jaringan yang memastikan bahwa data yang diterima oleh penerima adalah benar-benar sama dengan data yang dikirim oleh pengirim. Sebagai contoh *Alice* ingin mengirim pesan *M* ke *Bob* maka layanan keutuhan data memberikan pengetahuian kepada *Bob* bila *M* berubah.

5. *Non-Repudiation*

   Layanan *non-repudiation* adalah layanan keamanan jaringan yang menghindari penolakan atas penerimaan atau pengiriman data yang telah terkirim. Misalnya *Alice* mengirim pesan *M* ke *Bob*, maka *Bob* dengan layanan ini dapat memberi bukti bahwa data terkirim oleh *Alice* dan sebaliknya *Alice* dengan layanan yang sama dapat membuktikan bahwa pesan telah terkirim ke *Bob*.

6. Ketersediaan

   Layanan ketersediaan adalah layanan sistem yang membuat sumber daya sistem tetap dapat diakses dan digunakan ketika ada permintaan dari pihak yang

berwenang. Serangan terhadap sistem seperti *denial of services* membuat sistem tidak dapat diakses oleh pihak yang berwenang [4].

2.3.6 Database - MySQL

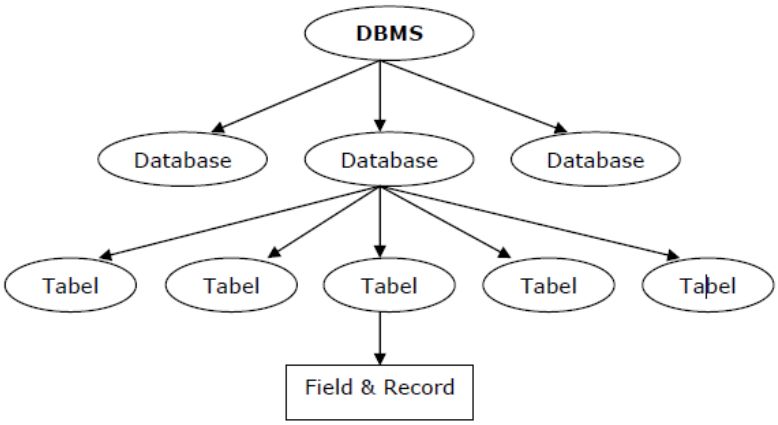
2.3.6.1 Pengenalan Database

Basis data (atau database) adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Database digunakan untuk menyimpan informasi atau data yang terintegrasi dengan baik di dalam komputer. Untuk mengelola *database* diperlukan suatu perangkat lunak yang disebut DBMS (Database Management System). DBMS merupakan suatu sistem perangkat lunak yang memungkinkan user (pengguna) untuk membuat, memelihara, mengontrol, dan mengakses database secara praktis dan efisien. Dengan DBMS, user akan lebih mudah mengontrol dan memanipulasi data yang ada. Sedangkan RDBMS atau Relationship Database Management System merupakan salah satu jenis DBMS yang mendukung adanya relationship atau hubungan antar tabel. Di samping RDBMS, terdapat jenis DBMS lain, misalnya Hierarchy DBMS, Object Oriented DBMS, dsb. Beberapa Istilah Database diantaranya:

1. Table: Sebuah tabel merupakan kumpulan data (nilai) yang diorganisasikan ke dalam baris (record) dan kolom (field). Masing-masing kolom memiliki nama yang spesifik dan unik.
2. Field: Field merupakan kolom dari sebuah table. Field memiliki ukuran type data tertentu yang menentukan bagaimana data nantinya tersimpan.
3. Record: merupakan sebuah kumpulan nilai yang saling terkait.
4. Key: Key merupakan suatu field yang dapat dijadikan kunci dalam operasi tabel. Dalam konsep database, key memiliki banyak jenis diantaranya Primary Key, Foreign Key, Composite Key, dll.
5. SQL: SQL atau Structured Query Language merupakan suatu bahasa (language) yang digunakan untuk mengakses database. SQL sering disebut juga sebagai query.

2.3.6.2 Hierarki Database

Dalam konsep database, urutan atau hierarki database sangatlah penting. Urutan atau hierarki database digambarkan dalam gambar sbb :



Gambar 2.5 Hierarki Database

2.3.6.3 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (bahasa Inggris: *database management system*) atau DBMS yang multithread, multi-user, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis di bawah lisensi GNU *General Public License* (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL. Tidak seperti PHP atau Apache yang merupakan software yang dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh penulisnya masing-masing, MySQL dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia yaitu MySQL AB. MySQL AB memegang penuh hak cipta hampir atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan MySQL AB adalah: David Axmark, Allan Larsson, dan Michael **"Monty"** Widenius.

2.3.6.4 Fitur-fitur MySQL

MySQL memiliki beberapa fitur penting diantaranya:

1. Relational Database System. Seperti halnya software database lain yang ada di pasaran, MySQL termasuk RDBMS.
2. Arsitektur Client-Server. MySQL memiliki arsitektur client-server dimana server database MySQL terinstal di server. Client MySQL dapat berada di komputer yang sama dengan server, dan dapat juga di komputer lain yang berkomunikasi dengan server melalui jaringan bahkan internet.
3. Mengenal perintah SQL standar. SQL (Structured Query Language) merupakan suatu bahasa standar yang berlaku di hampir semua software database. MySQL mendukung SQL versi SQL:2003.
4. Mendukung Sub Select. Mulai versi 4.1 MySQL telah mendukung select dalam select (sub select).
5. Mendukung Views. MySQL mendukung views sejak versi 5.0
6. Mendukung Stored Prosedured (SP). MySQL mendukung SP sejak versi 5.0
7. Mendukung Triggers. MySQL mendukung trigger pada versi 5.0 namun masih terbatas. Pengembang MySQL berjanji akan meningkatkan kemampuan trigger pada versi 5.1.
8. Mendukung replication.
9. Mendukung transaksi.
10. Mendukung foreign key. Tersedia fungsi GIS.
11. Free (bebas didownload)
12. Stabil dan tangguh
13. Fleksibel dengan berbagai pemrograman
14. Security yang baik
15. Dukungan dari banyak komunitas
16. Perkembangan software yang cukup cepat [9].

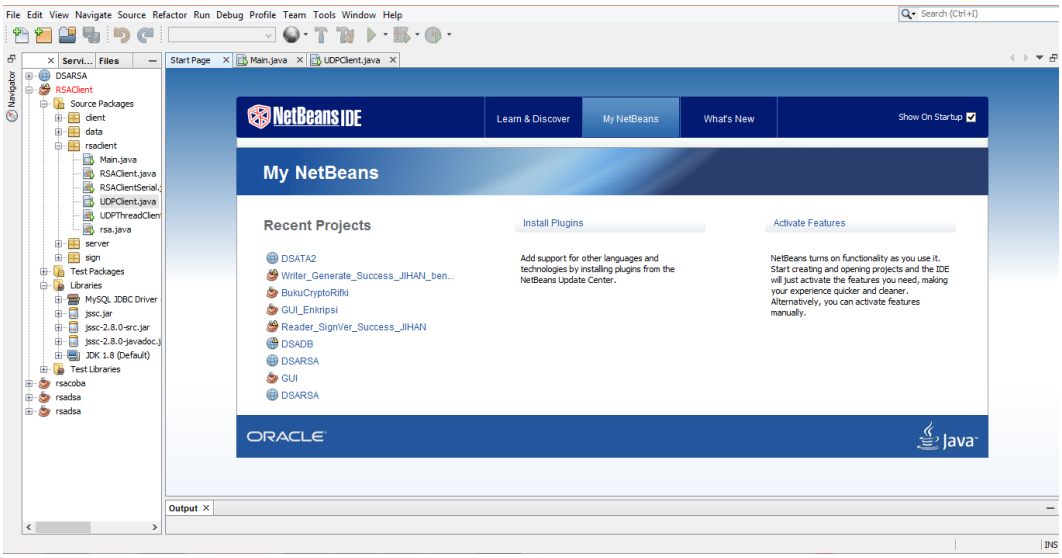
2.3.7 Netbeans

NetBeans merupakan sebuah proyek kode terbuka yang sukses dengan pengguna yang sangat luas, komunitas yang terus tumbuh, dan memiliki hampir

100 mitra (dan terus bertambah!). Sun Microsystems mendirikan proyek kode terbuka NetBeans pada bulan Juni 2000 dan terus menjadi sponsor utama. Saat ini terdapat dua produk : NetBeans IDE dan NetBeans Platform. NetBeans IDE adalah sebuah lingkungan pengembangan - sebuah kakas untuk pemrogram menulis, mengompilasi, mencari kesalahan dan menyebarkan program. Netbeans IDE ditulis dalam Java - namun dapat mendukung bahasa pemrograman lain. Terdapat banyak modul untuk memperluas Netbeans IDE. Netbeans IDE adalah sebuah produk bebas dengan tanpa batasan bagaimana digunakan.

Tersedia juga NetBeans Platform, yaitu sebuah fondasi yang modular dan dapat diperluas yang dapat digunakan sebagai perangkat lunak dasar untuk membuat aplikasi desktop yang besar. Mitra ISV menyediakan plug-in bernilai tambah yang dapat dengan mudah diintegrasikan ke dalam Platform dan dapat juga digunakan untuk membuat kakas dan solusi sendiri.

Kedua produk adalah kode terbuka (open source) dan bebas (free) untuk penggunaan komersial dan non komersial. Kode sumber tersedia untuk guna ulang dengan lisensi Common Development and Distribution License (CDDL) [10].

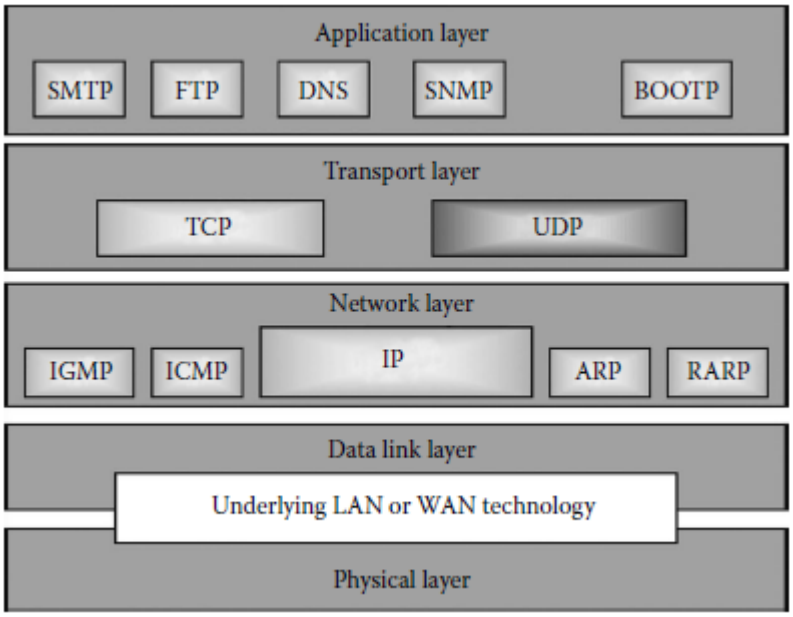


Gambar 2.6 Netbeans IDE

2.3.8 User Datagram Protocol

User Datagram Protocol (UDP) adalah bagian dari rangkaian TCP / IP. UDP menyediakan layanan penuh layer transport ke aplikasi. Juga memilik layer

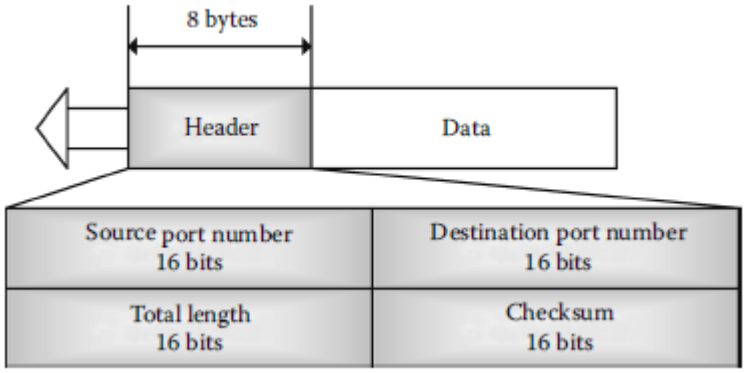
transport dalam model rangkaian TCP / IP, seperti yang ditunjukkan pada Gambar 2.7. UDP menyediakan koneksi antara dua proses di kedua ujung transmisi. Koneksi ini disediakan dengan overhead minimal, tanpa flow control atau acknowledge dari data yang diterima. Kontrol kesalahan minimal disediakan dengan mengabaikan paket yang diterima yang gagal tes checksum. UDP distandarisasi oleh Internet Society, yang merupakan organisasi independen internasional. Informasi lengkap tentang standar terkait UDP diterbitkan oleh RFC Editor [RFC].



Gambar 2.7 UDP dan TCP/IP pada model rangkaian TCP/IP

2.3.8.1 Datagram UDP

Datagram memiliki header 8 byte berukuran konstan yang ditambahkan ke data yang dikirimkan, seperti yang ditunjukkan pada Gambar 2.8.



Gambar 2.8 Format User Datagram

Pengertian dari setiap kolom header dijelaskan di bawah ini:

1. Alamat port sumber: memiliki panjang 16 bit yang berisi nomor port dari proses yang dikirim opsi atau data pada segmen.
2. Alamat port tujuan: memiliki panjang 16 bit yang berisi nomor port dari proses menerima opsi atau data yang dibawa oleh segmen.
3. Panjang total: memiliki panjang 16 bit yang berisi total panjang paket. Meskipun jumlahnya bisa berkisar dari 0 hingga 65.535, panjang minimumnya adalah 8 byte yang sesuai dengan paket dengan header dan tidak ada data. Panjang maksimum adalah 65,507 karena 20 byte digunakan oleh header IP dan 8 byte oleh header UDP. Dengan demikian, informasi ini redundan dengan panjang paket yang disimpan dalam header IP. Perpanjangan ke UDP yang memungkinkan untuk pengiriman datagram yang lebih besar melalui paket IPv6 telah distandarkan.
4. Checksum: memiliki panjang 16 bit yang berisi checksum.
5. Verifikasi checksum: penerima menghitung checksum baru untuk paket yang diterima yang berisi checksum original, setelah menambahkan pseudoheader seperti pada gambar 3. Jika checksum yang baru berupa nonzero, maka datagram rusak dan dibuang.

2.3.8.2 Penetapan Nomor Port

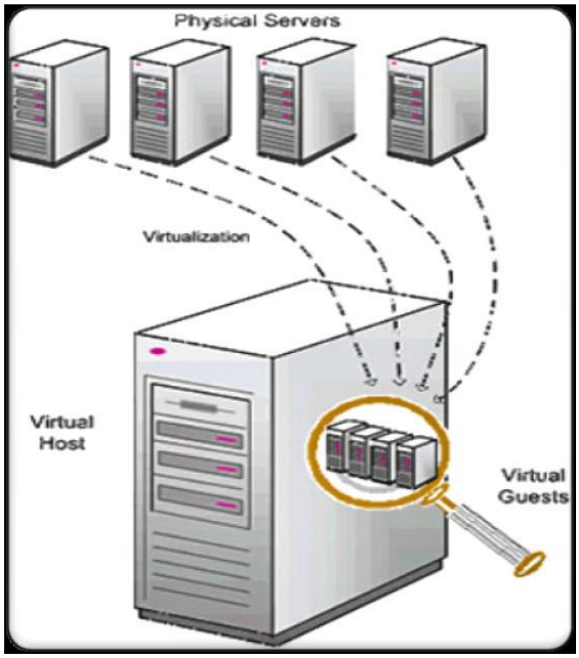
Untuk menyediakan platform untuk proses pengalamatan independent pada sebuah host, masing-masing koneksi dari proses ke jaringan diberi 16 bit. Ada tiga kategori nomor port, yaitu umum (0-1023), terdaftar (1024-49151), dan ephemeral (49152-6553). Port umum digunakan secara umum. Beberapa sistem operasi mengharuskan proses yang menggunakan port ini harus memiliki hak administratif. Persyaratan ini secara historis dibuat untuk menghindari hacker menjalankan pembajakan server pada sistem multi-user. Port umum ini terdaftar dan dikendalikan oleh Internet Assigned Numbers Authority. Port terdaftar juga terdaftar oleh IANA untuk menghindari kemungkinan konflik di antara berbagai aplikasi yang mencoba menggunakan port yang sama untuk mendengarkan koneksi yang masuk. Port Ephemeral juga disebut port dinamis. Port ini digunakan oleh koneksi keluar yang biasanya ditetapkan ke port pertama yang tersedia di atas 49.151. Beberapa sistem operasi mungkin tidak mengikuti rekomendasi IANA dan memperlakukan rentang port yang terdaftar sebagai singkat. Misalnya, BSD menggunakan port 1.024 hingga 4.999 sebagai port ephemeral, banyak kernel Linux menggunakan 32.768–61.000, Windows menggunakan 1.025–5.000, sementara FreeBSD mengikuti port IANA [11].

2.3.9 Virtual Private Server (VPS)

Virtualisasi merupakan suatu aplikasi perangkat lunak untuk mensimulasikan sumber daya perangkat keras. Menurut Sundarranjan, virtualisasi adalah sebuah teknik agar perangkat keras pada sebuah mesin dapat dibagi melalui pembagian perangkat keras atau lunak, berbagi waktu dan simulasi menjadi banyak lingkungan eksekusi, tiap bagian dapat berperan sebagai sistem yang lengkap. Sumber daya perangkat keras dibagikan di antara klien-klien yang berpikir bahwa mereka berjalan di atas perangkat keras asli [12].

VPS (*Virtual Private Server*) adalah server yang memiliki independensi dalam pengeloaan sumber daya server tanpa mitra. Diadministrasi secara virtual. Pelanggan dapat memiliki kontrol total dari server pribadinya dan dapat melakukan apapun yang pelanggan inginkan. Berbeda dari shared server, dimana pelanggan harus mengikuti syarat dan peraturan dari pusat , dan bahkan tidak dapat

menjalankan program yang dimiliki pelanggan. Sehingga VPS adalah solusi terbaik untuk menyediakan server independen dengan biaya yang terbatas dan juga terpercaya.



Gambar 2.9 Virtual Private Server

Skema VPS pada dasarnya bekerja dengan memiliki banyak jenis server web virtual diatas server web fisik. VPS hanya berbagi sejumlah sumber daya dari server web fisik. Setiap VPS diberikan fitur-fitur khusus untuk setiap sumber daya yang digunakan. Hasilnya, server web VPS lain yang sibuk atau kelebihan beban tidak dapat mempengaruhi server web lainnya. Properti ini dianggap sebagai fitur paling penting dari efisiensi VPS.

Sistem virtualisasi memberikan setiap VPS *resource* yang tetap. VPS server hanya dapat menggunakan yang diberikan. Ketika server VPS sedang kelebihan beban, hanya server yang kelebihan beban tersebut yang akan *down* tanpa merusak server VPS lain yang berbagi server fisik yang sama.

Hal yang menakjubkan dalam sistem virtualisasi adalah bahwa kebanyakan sistem operasi dapat dijalankan tidak peduli perangkat keras apa yang dimiliki server fisik. Sistem virtualisasi juga memiliki kemampuan untuk mengubah, menyalin, membuat cadangan/ *backup*, dan memodifikasi server VPS [13].

2.3.10 Arduino

Arduino dikatakan sebagai sebuah *platform* dari *physical computing* yang bersifat *open source*. Kata "platform" adalah sebuah pilihan kata yang tepat. Arduino tidak hanya sekedar sebuah alat pengembangan, tetapi ia adalah kombinasi dari hardware, bahasa pemrograman dan Integrated Development Environment (IDE) yang canggih.

IDE adalah sebuah software yang sangat berperan untuk menulis program, meng-*compile* menjadi kode biner dan meng-*upload* ke dalam *memory* microcontroller. Ada banyak projek dan alat-alat dikembangkan oleh akademisi dan profesional dengan menggunakan Arduino, selain itu juga ada banyak modul-modul pendukung (sensor, tampilan, penggerak dan sebagainya) yang dibuat oleh pihak lain untuk bisa disambungkan dengan Arduino. Arduino berevolusi menjadi sebuah platform karena ia menjadi pilihan dan acuan bagi banyak praktisi.

Salah satu yang membuat Arduino memikat hati banyak orang adalah karena sifatnya yang open source, baik untuk hardware maupun software-nya. Diagram rangkaian elektronik Arduino digratiskan kepada semua orang bebas men-download gambarnya, membeli komponen-komponennya, membuat PCB-nya dan merangkainya sendiri tanpa harus membayar kepada para pembuat Arduino. Sama halnya dengan IDE Arduino yang bisa di-download dan diinstal pada komputer secara gratis.

Secara umum Arduino terdiri dari dua bagian, yaitu hardware: papan input/output (I/O), software: Software Arduino meliputi IDE untuk menulis program, driver untuk koneksi dengan komputer, contoh program dan library untuk pengembangan program.

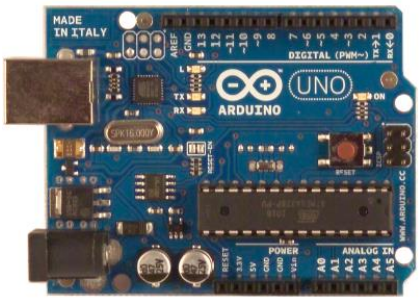
2.3.10.1 Jenis-jenis Papan Arduino

Saat ini ada bermacam-macam bentuk papan Arduino yang disesuaikan dengan peruntukannya seperti diperlihatkan berikut ini:

1. Arduino USB

Arduino USB menggunakan USB sebagai antar muka pemrograman atau komunikasi komputer. Contohnya adalah arduino uno, arduino duemilanove,

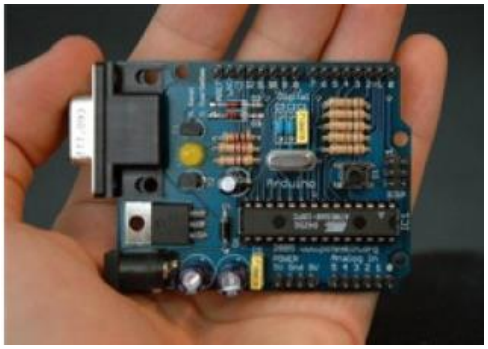
arduino diecimila, arduino NG Rev. C, arduino NG (Nuova Generazione), arduino Extreme dan arduino Extreme v2, arduino USB dan Arduino USB v2.0.



Gambar 2.10 Arduino USB

2. Arduino Serial

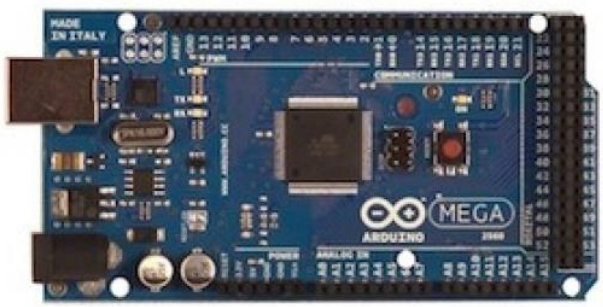
Menggunakan RS232 sebagai antar muka pemrograman atau komunikasi komputer. Contoh arduino serial dan arduino serial v2.0.



Gambar 2.11 Arduino Serial

3. Arduino Mega

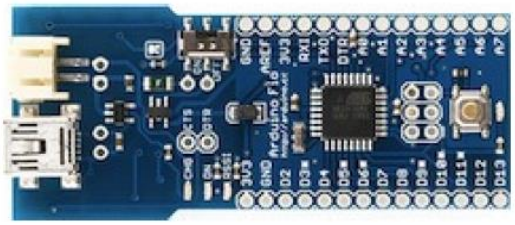
Papan Arduino dengan spesifikasi yang lebih tinggi, dilengkapi tambahan pin digital, pin analog, port serial dan sebagainya. Contohnya adalah Arduino Mega, Arduino Mega 2560.



Gambar 2.12 Arduino Mega

4. Arduino Fio

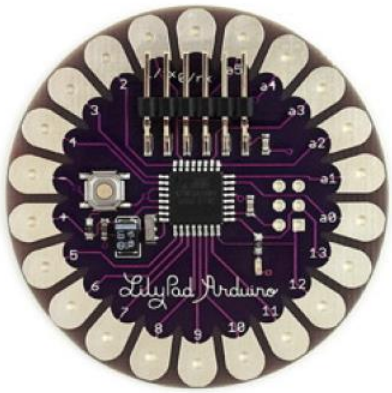
Arduino fio ditujukan untuk penggunaan *wireless* atau nirkabel.



Gambar 2.13 Arduino Fio

5. Arduino Lilypad

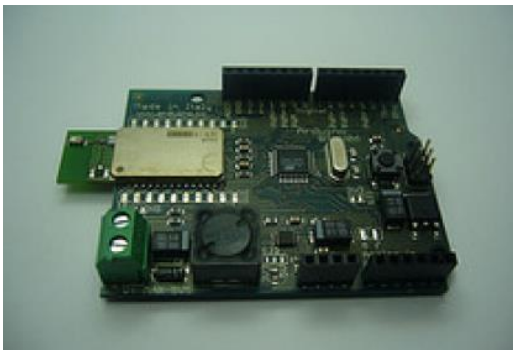
Papan dengan bentuk yang melingkar. Contoh: LilyPad Arduino 00, LilyPad Arduino 01, LilyPad Arduino 02, LilyPad Arduino 03, LilyPad Arduino 04.



Gambar 2.14 Arduino Lilypad

6. Arduino Bluetooth

Arduino BT mengandung modul bluetooth untuk komunikasi nirkabel.



Gambar 2.15 Arduino Bluetooth

7. Arduino Nano dan Arduino Mini

Papan berbentuk kompak dan digunakan bersama breadboard. Contoh : Arduino nano 3.0, Arduino nano 2.x , adruino mini 04, Arduino mini 03, arduino stamp 02 [14].

Gambar 2.16 Arduino Nano

2.3.10.2 Pengenalan Arduino Mega 2560

Arduino Mega 2560 adalah sebuah papan mikrokontroler berbasis Atmega 2560 *(datasheet)*. Mempunyai 54 pin digital *input/output* (dimana 14 pun dapat diguanakan sebagai keluaran PWM), 16 pin input analog, 2 UARTs (*Hardware serial ports*), sebuah *crystal oscillator* 16 MHz, sebuah penghubung USB, sebuah colokan listrik, ICSP *header*, dan tombol kembali. Setiap isi dari Arduino Mega 2560 membutuhkan dukungan mikrokontroler; koneksi mudah antara Arduino mega 2560 ke komputer dengan sebuah kabel USB atau daya dengan AC to DC adaptor atau baterai untuk memulai. Arduino mega cocok sebagai rancangan pelindung untuk Arduino *Deumilanove* atau *Diecimila*.

2.3.10.3 Arsitektur Arduino Mega 2560

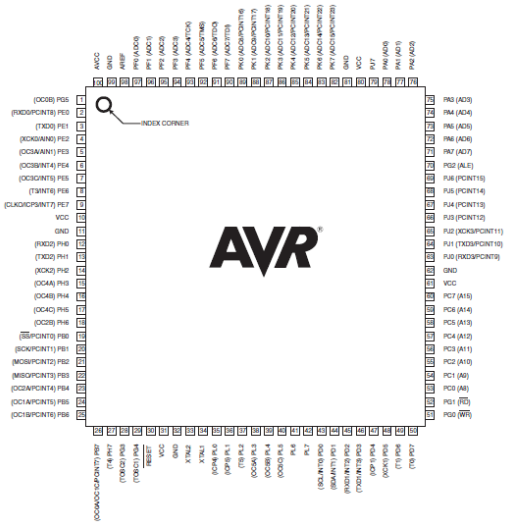
Arduino Mega 2560 terbentuk dari prosessor yang dikenal dengan Mikrokontroler ATMega 2560. Mikrokontroler ATMega 2560 memiliki beberapa fitur / spesifikasi yang menjadikannya sebagai solusi pengendali yang efektif untuk berbagai keperluan. Fitur-fitur tersebut antara lain :

1. Tegangan Operasi sebesar 5 V

2. Tegangan input sebesar 6 – 20 V tetapi yang direkomendasikan untuk ATMega 2560 sebesar 7 – 12 V.
3. Pin digital I/O sebanyak 54 pin dimana 14 pin merupakan keluaran dari PWM.
4. Pin input analog sebanyak 16 pin
5. Arus DC pin I/O sebesar 40 mA sedangkan Arus DC untuk pin 3.3V sebesar 50 mA
6. *Flash memory* 156 Kb yang mana 8 Kb digunakan oleh bootloader.
7. SRAM 8 Kbyte
8. EEPROM 4 Kbyte
9. Serta mempunyai 2 Port UARTs untuk komunikasi serial.

Gambar 2.17 ATMega 2560 pada Arduino Mega 2560

2.3.10.4 Konfigurasi Pin Arduino Mega



Gambar 2.18 Konfigurasi Pin Arduino Mega

1. VCC adalah tegangan catu digital
2. GND adalah Ground
3. Port A (PA7..PA0)

Port A adalah sebuah port I/O 8 bit dua arah dengan internal pull-up resistor (dipilih untuk masing-masing bit). Penyangga output Port A memiliki karakter penggerak karakteristik dengan kedua sink tinggi dan kemampuan sumber. Sebagai input, pin Port A eksternal pulled low sumber arus jika resistor pull-up aktif. Pin port A dinyatakan tri ketika sebuah kondisi reset menjadi aktif, bahkan jika waktu tidak berjalan. Port A juga menyajikan fungsi dari berbagai fitur spesial dari Atmega640/1280/1281/2560/2561.

4. Port B (PB7..PB0)

Port B adalah sebuah port I/O 8 bit dua arah dengan internal pull-up resistor (dipilih untuk masing-masing bit). Penyangga output Port B memiliki karakter penggerak karakteristik dengan kedua sink tinggi dan kemampuan sumber. Sebagai input, pin Port A eksternal pulled low sumber arus jika resistor pull-up aktif. Pin port A dinyatakan tri ketika sebuah kondisi reset menjadi aktif, bahkan jika waktu tidak berjalan. Port B empunyai kemampuan bergerak lebih baik daripada port lainnya.

5. Port C (PC7..PC0)

Port C adalah sebuah port I/O 8 bit dua arah dengan internal pull-up resistor (dipilih untuk masing-masing bit). Penyangga output Port C memiliki karakter penggerak karakteristik dengan kedua sink tinggi dan kemampuan sumber. Sebagai input, pin Port C eksternal pulled low sumber arus jika resistor pull-up aktif. Pin port C dinyatakan tri ketika sebuah kondisi reset menjadi aktif, bahkan jika waktu tidak berjalan.

6. Port D (PD7..PD0)

Port D adalah sebuah port I/O 8 bit dua arah dengan internal pull-up resistor (dipilih untuk masing-masing bit). Penyangga output Port D memiliki karakter penggerak karakteristik dengan kedua sink tinggi dan kemampuan sumber. Sebagai input, pin Port D eksternal pulled low sumber arus jika resistor pull-up aktif. Pin port D dinyatakan tri ketika sebuah kondisi reset menjadi aktif, bahkan jika waktu tidak berjalan.

7. Port E (PE7..PE0)

Port E adalah sebuah port I/O 8 bit dua arah dengan internal pull-up resistor (dipilih untuk masing-masing bit). Penyangga output Port E memiliki karakter penggerak karakteristik dengan kedua sink tinggi dan kemampuan sumber. Sebagai input, pin Port E eksternal pulled low sumber arus jika resistor pull-up aktif. Pin port E dinyatakan tri ketika sebuah kondisi reset menjadi aktif, bahkan jika waktu tidak berjalan.

8. Port F (PF7..PF0)

Port F disajikan sebagai masukan analog ke A/D converter. Port F juga menyajikan sebuah port I/O 8 bit dua arah, jika A/D Converter tidak digunakan. Pin port dapat menyediakan internal pull-up resistor (dipilih untuk masing-masing bit). Penyangga output Port F memiliki karakter penggerak karakteristik dengan kedua sink tinggi dan kemampuan sumber. Sebagai input, pin Port F eksternal pulled low sumber arus jika resistor pull-up aktif. Pin port F dinyatakan tri ketika sebuah kondisi reset menjadi aktif, bahkan jika waktu tidak berjalan. Jika antarmuka JTAG mengizinkan, pull-up resistor pada pin PF7(TDI), PF5(TMS), dan PF4(TCK) akan iaktifkan bahkan jika terjadi reset. Port F juga menyajikan fungsi dari antarmuka JTAG.

9. Port G (PG7..PG0)

Port G adalah sebuah port I/O 6 bit dua arah dengan internal pull-up resistor (dipilih untuk masing-masing bit). Penyangga output Port G memiliki karakter penggerak karakteristik dengan kedua sink tinggi dan kemampuan sumber. Sebagai input, pin Port G eksternal pulled low sumber arus jika resistor pull-up aktif. Pin port G dinyatakan tri ketika sebuah kondisi reset menjadi aktif, bahkan jika waktu tidak berjalan.

10. Port H (PH7..PH0)

Port H adalah sebuah port I/O 8 bit dua arah dengan internal pull-up resistor (dipilih untuk masing-masing bit). Penyangga output Port H memiliki karakter penggerak karakteristik dengan kedua sink tinggi dan kemampuan sumber. Sebagai input, pin Port H eksternal pulled low sumber arus jika resistor pull-up aktif. Pin port H dinyatakan tri ketika sebuah kondisi reset menjadi aktif, bahkan jika waktu tidak berjalan.

11. Port J (PJ7..PJ0)

Port J adalah sebuah port I/O 8 bit dua arah dengan internal pull-up resistor (dipilih untuk masing-masing bit). Penyangga output Port J memiliki karakter penggerak karakteristik dengan kedua sink tinggi dan kemampuan sumber. Sebagai input, pin Port J eksternal pulled low sumber arus jika resistor pull-up aktif. Pin port J dinyatakan tri ketika sebuah kondisi reset menjadi aktif, bahkan jika waktu tidak berjalan.

12. Port K (PK7..PK0)

Port K disajikan sebagai masukan analog ke A/D converter. Port K adalah sebuah port I/O 8 bit dua arah dengan internal pull-up resistor (dipilih untuk masing-masing bit). Penyangga output Port K memiliki karakter penggerak karakteristik dengan kedua sink tinggi dan kemampuan sumber. Sebagai input, pin Port K eksternal pulled low sumber arus jika resistor pull-up aktif. Pin port K dinyatakan tri ketika sebuah kondisi reset menjadi aktif, bahkan jika waktu tidak berjalan.

13. Port L (PL7..PL0)

Port L adalah sebuah port I/O 8 bit dua arah dengan internal pull-up resistor (dipilih untuk masing-masing bit). Penyangga output Port L memiliki karakter penggerak karakteristik dengan kedua sink tinggi dan kemampuan sumber. Sebagai input, pin Port L eksternal pulled low sumber arus jika resistor pull-up aktif. Pin port L dinyatakan tri ketika sebuah kondisi reset menjadi aktif, bahkan jika waktu tidak berjalan.

14. Reset

Input reset. Sebuah level rendah pada pin ini untuk lebih panjang dari pada panjang minimum pulsa akan menghasilkan sebuah reset, bahkan jika waktu tidak berjalan. Panjang minimum pulsa dijelaskan pada “Sistem dan karakter reset” pada halaman 360. Pulsa terpendek tidak dijamin menghasilkan sebuah reset .

15. XTAL1

Input ke inverting amplifier oscilator dan input ke internal jalur operasi waktu.

16. XTAL2

Keluaran dari inverting oscilator amplifier.

17. AVCC

AVCC merupakan pin tegangan catu untuk port F dan A/D Converter. AVCC dapat terhubung secara eksternal ke VCC, bahkan jika ADC tidak digunakan jika ADC digunakan, ADC akan terhubung ke VCC melalui sebuah low pass filter.

18. AREF

AREF adalah pin referensi analog untuk A/D Converter (Atmel Corporation.2014).

2.3.11 Metode Serangan Pasif

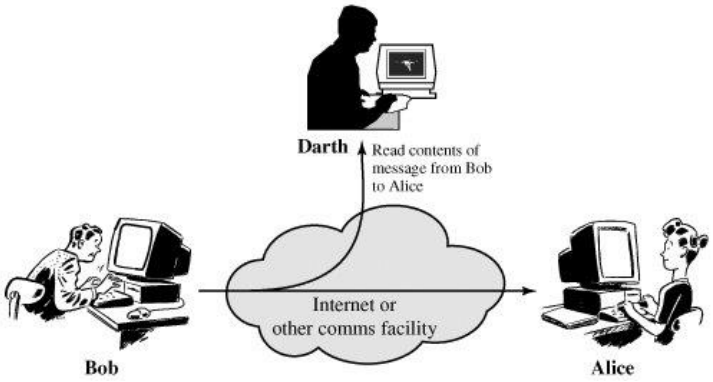
Serangan pasif adalah jenis serangan yang tidak membahayakan terhadap sebuah sistem jaringan. Jenis serangan ini tidak menyebabkan hilangnya sumber daya dalam sebuah jaringan maupun menyebabkan kerusakan terhadap sebuah sistem jaringan yang di serang menggunakan jenis serangan ini. Sumber daya yang terdapat dalam sistem jaringan diantaranya berupa data, bandwidth jaringan, printer, memori dalam sebuah komputer, unit pengolah (prosesor) dan masih banyak lagi. Intinya jenis serangan ini hanya melakukan pengamatan terhadap semua sumber daya yang terdapat dalam sebuah sistem jaringan komputer. Seperti memantau lalu lintas jaringan sebuah sistem jaringan komputer. Informasi yang dihasilkan dari hasil pengamatan tersebut sangat bermanfaat bagi pihak yang tidak berhak untuk melakukan penyerangan selanjutnya terhadap sistem tersebut. sehingga jenis serangan ini sangat sulit untuk di deteksi oleh pengelola sebuah sistem jaringan komputer.

Komunikasi jaringan tanpa kabel biasanya menggunakan frekuensi gelombang radio umum yang tidak terdaftar yang dapat di akses oleh siapapun dengan menggunakan kartu jaringan yang kompatibel, sehingga untuk jaringan jenis ini sangat mudah untuk di sadap dengan menggunakan teknik "*sniffing*" atau "*wardriving*". Saat ini banyak "*sniffe*r" menggunakan software seperti NetStumbler dengan kombinasi antena yang saling bekerja bersama dengan kartu jaringan tanpa kabel (*wireless*) untuk mendeteksi jaringan *"access point*" (AP) yang berada dalam jangkauan dan sinyalnya dapat diakses kartu jaringan tanpa kabel tersebut. Kemudian *traffic data* yang terjadi didalam jaringan wireless tersebut ditangkap oleh "*sniffe*r" tersebut untuk kemudian di analisis dengan menggunakan tool seperti Microsoft Network Monitor untuk sistem operasi microsoft windows atau menggunakan Linux TCP Dump untuk sistem operasi Linux.

Seranagan pasif merupakan serangan pada sistem autentikasi yang tidak menyisipkan data pada aliran data, tetapi hanya mengamati atau memonitor pengiriman informasi ke tujuan. Informasi ini dapat digunakan di lain waktu oleh pihak yang tidak bertanggung jawab. Serangan pasif yang mengambil suatu unit data kemudian menggunakannya untuk memasuki sesi autentikasi dengan berpura-pura menjadi user yangg autentik / asli disebut dengan replay attack. Beberapa informasi autentikasi seperti password atau data biometric yang dikirim melalui transmisi elektronik dapat direkam dan kemudian digunakan untuk memalsukan data yang sebenarnya. Serangan pasif ini sulit dideteksi karena penyerang tidak melakukan perubahan data. Oleh sebab itu untuk mengatasi serangan pasif ini lebih ditekankan pada pencegahan daripada pendeteksiannya. William Stalling menyatakan serangan pasif bertujuan adalah untuk mendapatkan informasi yang sedang di transmisikan. Ada dua jenis serangan pasif yaitu:

1. Release of Message Content

Release of message content adalah serangan yang bertujuan untuk mendapatkan informasi yang dikirim baik melalui percakapan telepon, email, ataupun transfer file dalam jaringan.

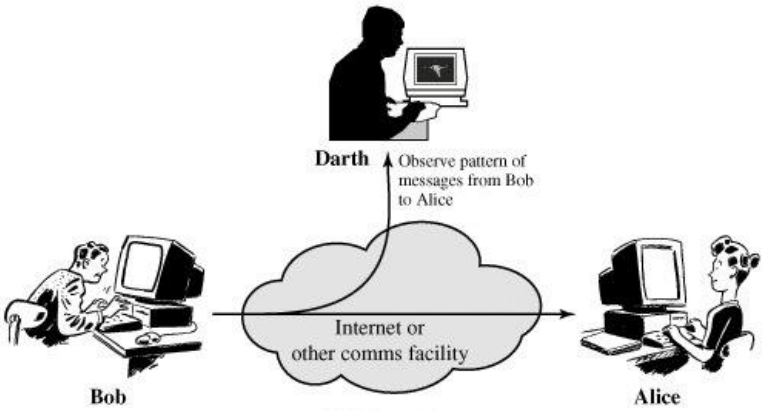


Gambar 2.19 Release of Message Content

2. Traffic Analysis

Traffic Analysis adalah suatu serangan yang dilakukan dengan mengambil informasi yang telah diberi proteksi (misal enkripsi) dan dikirimkan dalam jaringan, kemudian mencoba menganalisa sistem proteksi informasi tersebut untuk kemudian dapat memecahkan sistem proteksi informasi tersebut. Misalkan kita memiliki cara menutupi isi pesan atau informasi lalu lintas lain sehingga lawan, bahkan jika mereka menangkap pesan, tidak bisa mengekstrak informasi dari pesan. Teknik umum untuk isi masking adalah enkripsi. Jika kita memiliki perlindungan enkripsi, lawan masih mungkin bisa mengamati pola pesan-pesan. Lawan bisa menentukan lokasi dan identitas host yang berkomunikasi dan bisa mengamati frekuensi dan

panjang pesan-pesan yang dipertukarkan. Informasi ini mungkin berguna dalam menebak sifat dari komunikasi yang sedang berlangsung.

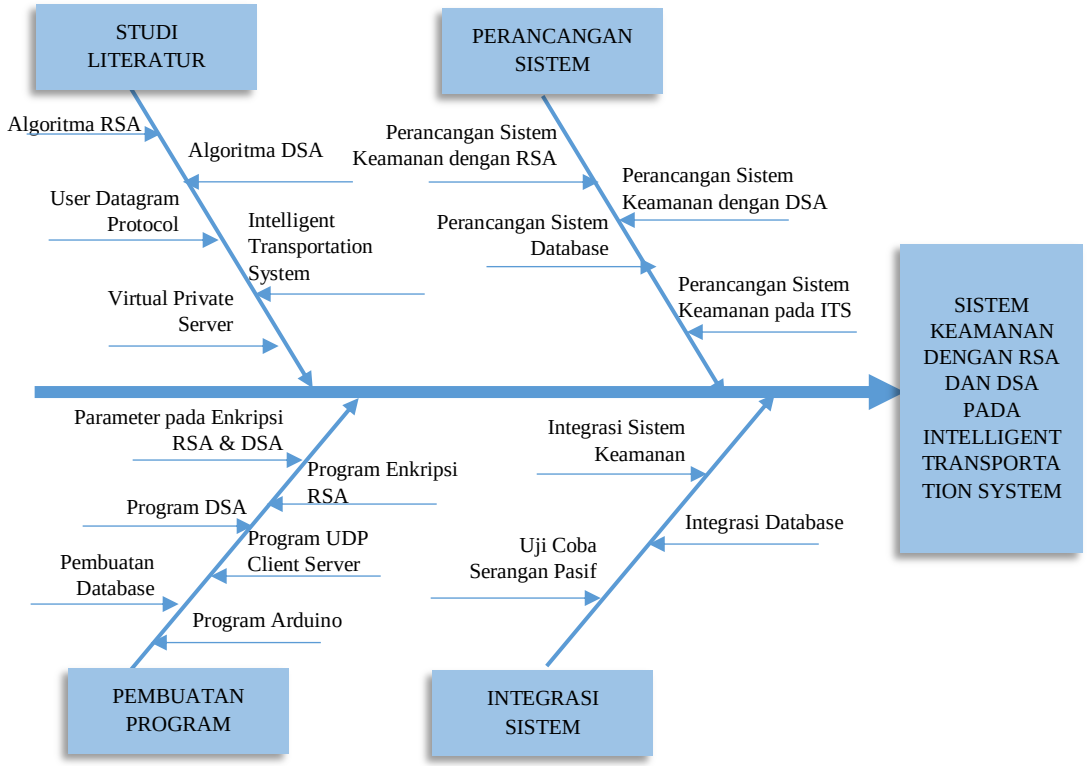


Gambar 2.20 Traffic Analysis [15].

BAB 3
METODOLOGI PENELITIAN

3.1 Langkah-Langkah Penelitian

Langkah-langkah yang dilakukan di dalam penelitian Tesis ini secara umum dijelaskan dengan menggunakan diagram *Fishbone* seperti yang ditunjukkan pada Gambar 3.1.



Gambar 3.1 Fishbone Perancangan Sistem

Penelitian diawali dengan studi literatur. Studi literatur diperoleh dari penelitian sebelumnya yang telah dilakukan peneliti sebelumnya yang digunakan sebagai pembanding kinerja dengan tesis yang akan dilakukan.

Setelah melakukan studi literatur, langkah selanjutnya adalah perancangan sistem. Perancangan sistem meliputi perancangan sistem keamanan dengan RSA, perancangan sistem keamanan dengan DSA, perancangan sistem database, dan perancangan sistem keamanan pada *Intelligent Transportation System* yaitu dengan menggabungkan kedua metode RSA dan DSA.

Langkah selanjutnya adalah pembuatan program. Sebelum melakukan pembuatan program, maka dilakukan pemilihan parameter-parameter apa saja yang akan digunakan pada sistem yang dibuat. Parameter-parameter yang maksud adalah parameter untuk algoritma enkripsi RSA dan *Digital Signature Algorithm.* Setelah itu barulah pembuatan program dimulai dengan pembuatan program enkripsi RSA, dan pembuatan program *Digital Signature Algorithm.* Setelah kedua program berhasil dijalankan, maka dilakukan penggabungan kedua metode RSA dan *Digital Signature Algorithm* guna meningkatkan level keamanan. Langkah selanjutnya adalah pembuatan program UDP client server, pembuatan program pada arduino, pembuatan sistem database.

Setelah semua program dibuat, maka hal terakhir yang dilakukan adalah integrasi dan uji coba sistem. Dalam penelitian ini, integrasi dilakukan dengan menggabungkan kedua metode RSA dan *Digital Signature Algorithm* untuk sistem keamanan dalam pertukaran informasi dari OBU ke server hingga TMC (*Traffic Management System)* dengan menggunakan protokol UDP. Server yang akan digunakan adalah menggunakan VPS (*Virtual Private Network*). Setelah semua sistem terintegrasi, maka dilakukan uji coba serangan dengan menggunakan passive attack. Dimana passive attack yang dilakukan adalah dengan melihat paket yang dikirim dengan menggunakan software wireshark.

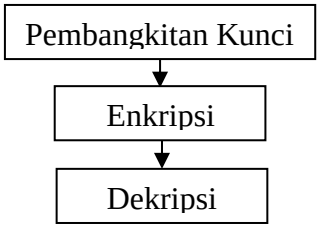
3.2 Perancangan Sistem

Pada sub bab perancangan sistem ini, terdapat beberapa bagian perancangan diantaranya perancangan sistem keamanan yang dalam penelitian ini menggunakan metode enkripsi RSA dan *Digital Signature Algorithm*, perancangan database dan yang terakhir adalah perancangan sistem keamanan pada *Intelligent Transportation System.*

3.2.1 Perancangan Sistem Keamanan

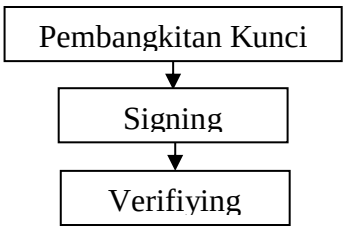
Sistem keamanan akan diterapkan pada sistem komunikasi *Intelligent Transport System* khususnya pada sistem komunikasi antara *On Board Unit* pada transportasi umum dengan menggunakan VPS (*Virtual Private Server*). Sistem keamanan yang digunakan adalah dengan metode gabungan enkripsi RSA dan

Digital Signature Algorithm. Berikut adalah proses pada metode enkripsi RSA.



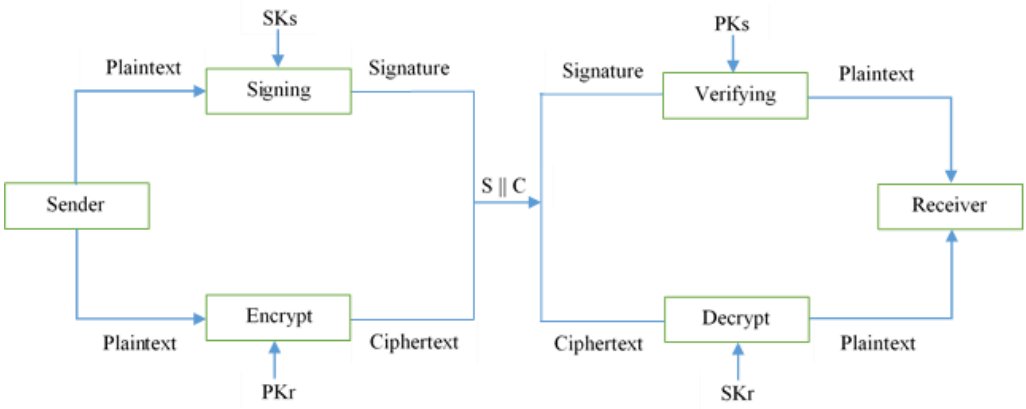
Gambar 3.2 Proses pada Enkripsi RSA

Metode enkripsi RSA memiliki tiga tahap yaitu, pembangkitan kunci, enkripsi, dan dekripsi. Berbeda dengan tahapan dari enkripsi, metode *Digital Signature Algorithm* memiliki tiga tahap yaitu, pembangkitan kunci, proses *signing*, dan proses *verifiying*.



Gambar 3.3 Proses pada *Digital Signature Algorithm*

Gambar 3.4 menunjukkan alur sistem keamanan dengan menggunakan metode enkripsi RSA dan *Digital Signature Algorithm* secara keseluruhan.



Gambar 3.4 Alur Sistem Keamanan dengan Metode RSA dan DSA

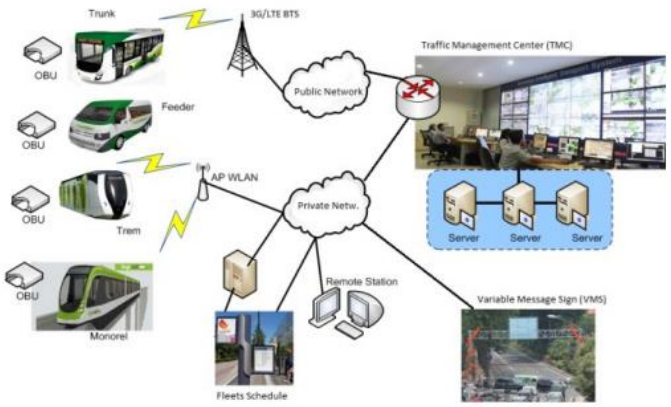
Pembangkitan kunci dilakukan secara independent pada masing-masing OBU. Pembangkitan kunci dilakukan dengan memasukkan nomor polisi kendaraan dan sistem akan langsung membangkitkan kunci. Setelah setiap kendaraan

memiliki kunci-kuncinya, maka tidak perlu melakukan pembangkitan kunci lagi. Perbaruan kunci akan dilakukan pada kondisi tertentu.

Dari alur sistem keamanan diatas, setelah semua proses pembangkitan kunci selesai dapat dilihat bahwa pengirim akan mengirim pesan dengan mengenkripsi pesan menggunakan kunci publik pengirim yang akan menjadi ciphertext dan memberi *sign* atau tanda tangan pesan menggunakan kunci privat pengirim yang akan menjadi signature. Setelah itu, signature dan ciphertext dikirim bersamaan pada penerima. Di sisi penerima, penerima akan mendekripsi pesan menggunakan kunci privat penerima yang akan mengembalikan pesan utuh atau plaintext dan memverifikasi signature menggunakan kunci publik pengirim. Jika signature valid, maka pesan yang diterima asli. Dan jika signature invalid, maka pesan yang diterima palsu sehingga pesan tidak dapat dibuktikan keasliannya.

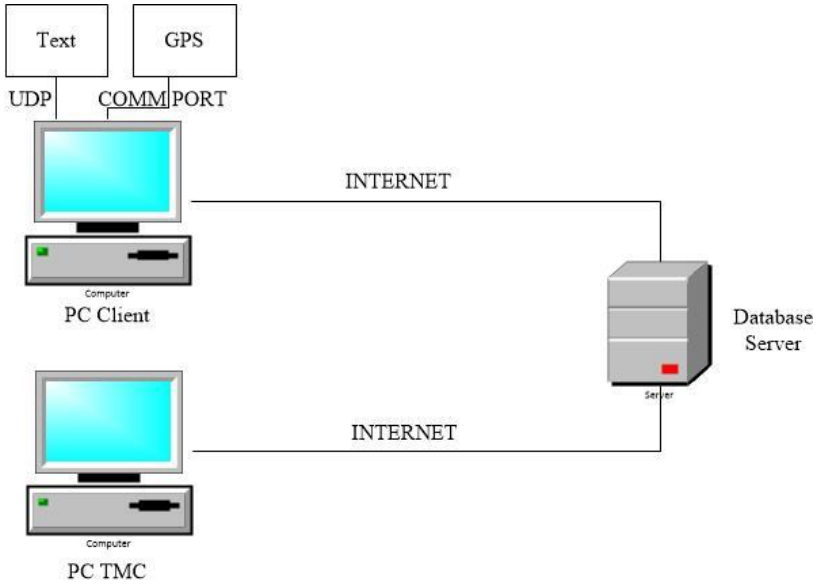
3.2.2 Perancangan Sistem Keamanan pada *Intelligent Transportation System*

Gambar 3.5 merupakan jaringan komunikasi global pada *Intelligent Transportation System* yang diusulkan. Terdapat OBU atau *On Board Unit* pada tiap-tiap transportasi umum. OBU akan mengirimkan beberapa informasi penting seperti lokasi, informasi lalu lintas, informasi komunikasi antara transportasi-transportasi tersebut dengan server, dan beberapa informasi penting lainnya. Informasi tersebut yang akan diberi sistem pengamanan dengan metode enkripsi RSA dan *Digital Signature Algorithm.*



Gambar 3.5 Jaringan Komunikasi ITS

Perancangan sistem keamanan pada *Intelligent Transportation System* adalah untuk membuat sebuah rancangan sistem yang akan mengintegrasi sistem keamanan dengan OBU. Dimana pada penelitian ini, pengiriman data lokasi diasumsikan tetap menggunakan arduino dengan komunikasi serial, dan pengiriman informasi berupa data text menggunakan UDP (*User Datagram Protocol*). Untuk metode enkripsi menggunakan enkripsi RSA dan digital signature menggunakan *Digital Signature Algorithm.* Metode sistem keamanan ini akan diterapkan pada sistem yang sudah ada pada OBU. Gambar 3.6 adalah blok diagram sederhana untuk perancangan sistem keamanan pada *Intelligent Transportation System.*



Gambar 3.6 Perancangan Sistem Keamanan pada ITS

Setelah data-data diterima oleh PC client dalam hal ini OBU, maka data-data tersebut akan dienkripsi dan diberi tanda tangan digital yang kemudian ciphertextnya akan dikirim ke database server melalui jaringan internet. Sedangkan PC TMC akan menerima ciphertext dari database server dan dapat sewaktu-waktu mendekripsi ciphertext tersebut.

3.2.3 Perancangan Sistem Database

Pada perancangan sistem database, database diberi nama ‘dbkunci’ dan memiliki tiga tabel. Tabel pertama diberi nama tabel ‘kendaraan’ dimana terdapat 2 kolom berisi IDKendaraan dan noPol. Tabel ‘kendaraan’ berfungsi untuk

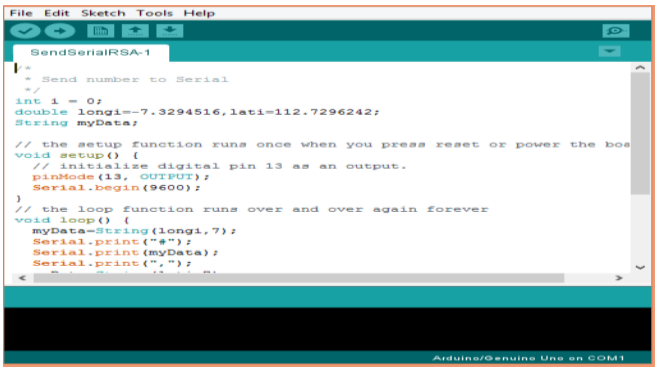
menyimpan id kendaraan dan nopor polisi kendaraan. Tabel kedua diberi nama ‘generatekey’ dimana terdapat beberapa kolom berisi IDKendaraan, p, q, g, x, y, e, d, n. Tabel ini berfungsi untuk menyimpan kunci atau nilai-nilai yang dibangkitkan pada waktu pembangkitan kunci. Tabel ketiga diberi nama ‘dataposition’ dimana terdapat 5 kolom berisi IDKendaraan, cyphertext, r, s, datetime. Tabel ‘dataposition’ berfungsi untuk menyimpan ciphertext dari data yang dienkripsi dan juga menyimpan signature dari proses signing seperti pada Tabel 3.1.

Tabel 3.1 Tabel Database

Kendaraan	Generatekey	Dataposition
IDKendaraan noPol	IDKendaraan P Q G X Y E D N	IDKendaraan Cyphertext R S Datetime

3.2.4 Perancangan Arduino untuk Data Lokasi

Perancangan simulasi pembuatan data lokasi menggunakan pemrograman pada arduino dengan membuat lokasi fix. Langkah pertama adalah menginstall software arduino IDE. Untuk hardware menggunakan arduino MEGA 2560 dan menggunakan komunikasi serial. Setelah software arduino IDE terinstall, langkah selanjutnya adalah membuat programnya dan mengcompile terlebih dahulu sebelum di *upload* ke hardware arduino MEGA 2560.



Gambar 3.7 Arduino IDE

3.3 Pembuatan Program

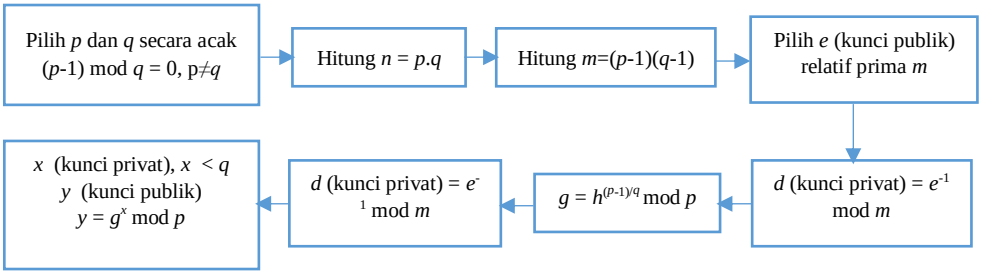
Pada sub bab pembuatan program, terdapat dua tahapan pembuatan program diantaranya pembuatan program sistem keamanan meliputi pembangkitan kunci, enkripsi signing, dan dekripsi verifying. Tahapan selanjutnya adalah pembuatan program client server menggunakan *User Datagram Protocol*. Dan yang terakhir adalah pembuatan program arduino untuk mengakses data lokasi.

3.3.1 Pembuatan Program Sistem Keamanan

Terdapat tiga proses pada metode enkripsi RSA yaitu pembangkitan kunci, enkripsi, dan dekripsi. Berikut adalah penjabaran dari masing-masing proses.

3.3.1.1 Pembangkitan Kunci

Pembangkitan kunci pada metode gabungan RSA dan *Digital Signature Algorithm* ini akan menghasilkan dua buah kunci privat dan dua buah kunci publik dengan algoritma seperti pada Gambar 3.8.



Gambar 3.8 Pembangkitan Kunci

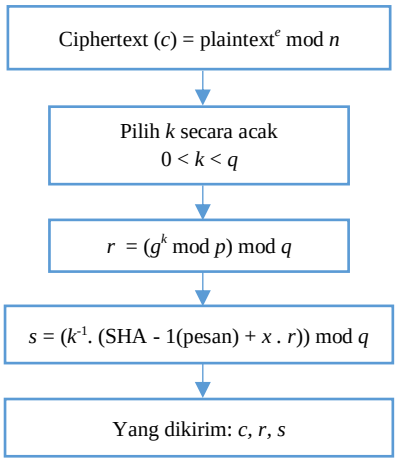
Untuk membangkitkan kunci seperti pada gambar diatas, maka program yang digunakan adalah seperti pada Gambar 3.9.

```
BigInteger p = new BigInteger(bitlen / 2, 100, r);
BigInteger q = new BigInteger(bitlen / 2, 100, r);
n = p.multiply(q);
BigInteger m =
     (p.subtract(BigInteger.ONE)).multiply(q.subtrac
     t(BigInteger.ONE));
e = new BigInteger("3");
   while (m.gcd(e).intValue() > 1) {
        e = e.add(new BigInteger("2"));
        }
        d = e.modInverse(m);
primeCenterie = 20;
q = new BigInteger(160, primeCenterie, sr);
        do {
        x = new BigInteger(160, sr);
        } while (x.compareTo(BigInteger.ZERO) != 1
&& x.compareTo(q) != -1);
p = generateP(q, 512);
g = generateG(p, q);
y = g.modPow(x, p);
```

Gambar 3.9 Program Pembangkitan Kunci

3.3.1.2 Enkripsi dan Signing

Setelah proses pembangkitan kunci, langkah selanjutnya adalah proses enkripsi dan proses signing. Gambar 3.10 dibawah ini adalah proses dari enkripsi dan signing.



Gambar 3.10 Proses Enkripsi dan Signing

Untuk melakukan enkripsi dan signing seperti pada gambar diatas, maka program yang digunakan adalah seperti pada Gambar 3.11.

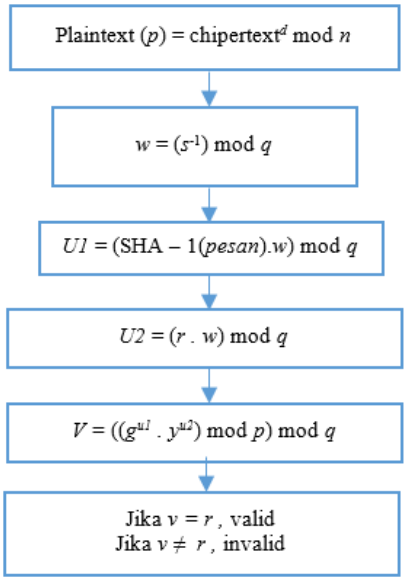
```
public synchronized String encrypt(String message)
     { return (new
     BigInteger(message.getBytes())).modPow(e,
     n).toString();

public BigInteger[] sign(String pesan, BigInteger
sk, BigInteger p, BigInteger q, BigInteger g)
     {rs[2] = sk;
     hshm = hshcall(pesan);
     k = generateK(q);
     r = g.modPow(k, p).mod(q);
     s = (k.modInverse(q).multiply(hshm.add
     (sk.multiply(r)))).mod(q);
     rs[0] = r;
```

Gambar 3.11 Program Enkripsi dan Signing

Pada proses enkripsi, plaintext akan difragmentasi setiap 128 karakter agar dapat melakukan proses enkripsi. Jika plaintext memiliki 300 karakter maka plaintext akan dipotong menjadi 3 bagian yaitu 128, 128, 44 karakter. Setiap bagian akan dienkripsi dan akan digabungkan kembali dalam bentuk ciphertext dan siap dikirimkan ke database server.

3.3.1.3 Dekripsi dan Verifying

Setelah proses enkripsi dan signing, langkah selanjutnya adalah proses dekripsi dan verifying. Gambar 3.12 dibawah ini adalah proses dari dekripsi dan verifying.



Gambar 3.12 Proses Dekripsi dan Verifying

Untuk melakukan dekripsi dan verifying seperti pada gambar diatas, maka program yang digunakan adalah seperti pada Gambar 3.13.

```
public synchronized String dencrypt(String message)
     { return new String((new BigInteger(message)).
     modPow(d, n).toByteArray());

int Sinpcmp = s.compareTo(q);
int Rinpcmp = rs[0].compareTo(q);
if (Sinpcmp == -1 && Rinpcmp == -1) {
     W = s.modInverse(q);
     U1 = (hshm1.multiply(W)).mod(q);
     U2 = (rs[0].multiply(W)).mod(q);
     Vint1 = (g.modPow(U1, p));
     Vint2 = rs[3].modPow(U2, p);
     Vint3 = Vint1.multiply(Vint2);
     Vint4 = Vint3.mod(p);
     V = Vint4.mod(q);
     int Vcmp = V.compareTo(rs[0]);
     if (Vcmp == 0) {
          return "Valid Signature";
          } else {return "Invalid Signature";}
        } else {return "Invalid Signature1";}
```

Gambar 3.13 Program Dekripsi dan Verifying

3.3.2 Pembuatan Program Client Server UDP

Pada pemrograman client server menggunakan *User Datagram Protocol* ini, dibuat program untuk client dan program untuk server. Keduanya harus menggunakan nomor port yang sama. Pada penelitian ini, menggunakan nomor port 1337. Program pada client adalah:

```
client = new ThreadedUDPClient("localhost", 1337);
```

sedangkan program pada server adalah:

```
server = new ThreadedUDPServer(1337);
```

3.3.3 Pembuatan Program Arduino

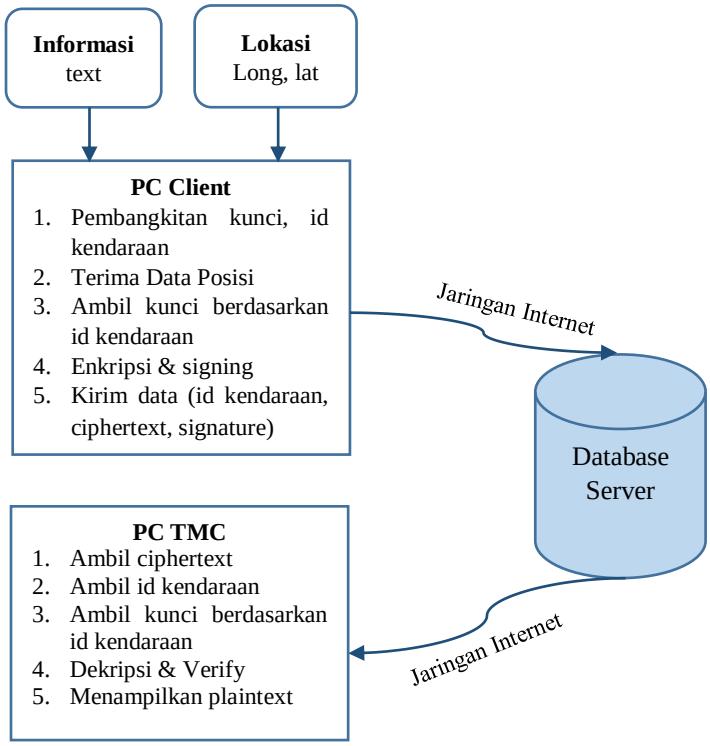
Pemrograman ada arduino dilakukan untuk pengganti gps, dalam artian arduino akan mengirimkan longitude dan latitude dan pengiriman dilakukan dalam waktu 5 detik sekali. Gambar 3.14 adalah listing program yang diupload pada arduino. Program tersebut menunjukkan nilai longitude dan latitude yang sudah didefinisikan diawal program. Dan dikirim setiap 5 detik atau 5000 mili detik sekali

```
int i = 0;
double longi=-7.3294516,lati=112.7296242;
String myData;
void loop() {
  myData=String(longi,7);
  Serial.print("#");
  Serial.print(myData);
  Serial.print(",");
  myData=String(lati,7);
  Serial.print(myData);
  Serial.println(";");
  i++;

  digitalWrite(13, HIGH
  delay(5000);
  digitalWrite(13, LOW
  delay(1000);
}
```

Gambar 3.14 Program Data Lokasi pada Arduino

3.4 Integrasi Sistem

Integrasi sistem dilakukan untuk menggabungkan antara rancangan sistem keamanan pada sistem komunikasi *Intelligent Transportation System.* Gambar 3.15 menunjukkan blok diagram integrasi sistem yang dilakukan pada penelitian ini.



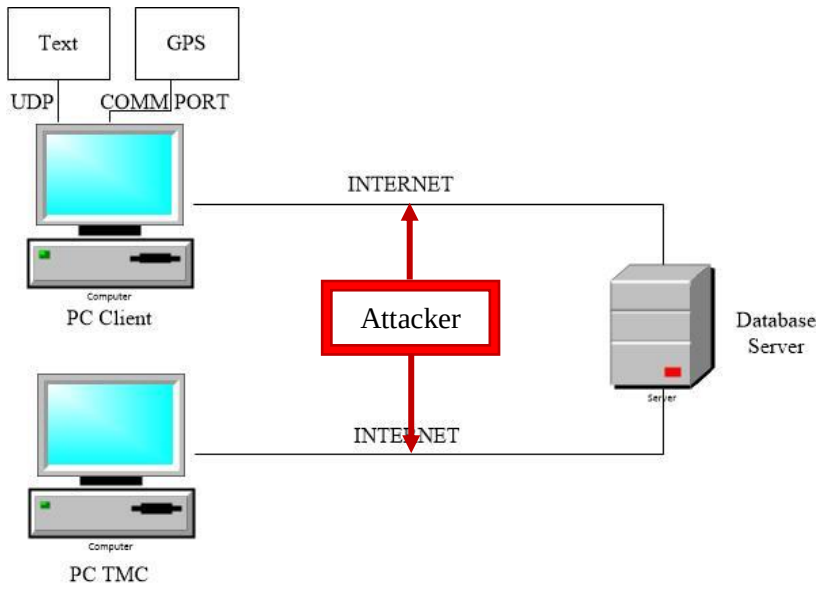
Gambar 3.15 Integrasi Sistem Keseluruhan

Sesuai gambar diatas, OBU akan mengirimkan informasi berupa text dan juga data lokasi. Dalam penelitian ini, data lokasi diasumsikan menggunakan arduino. Data lokasi yang dikirim berupa longitude dan latitude. Informasi yang berupa text dikirim melalui protokol UDP, sedangkan data lokasi dikirim menggunakan komunikasi serial. Pada PC Client, pertama dilakukan pembangkitan kunci dan id kendaraan. Setelah itu, PC dapat menerima data posisi dan data text. Untuk proses enkripsi dan signing, dibutuhkan beberapa parameter yang ada pada database. Sehingga sebelumnya PC client akan mengambil data dari database berdasarkan id kendaraan. Setelah dienkripsi dan signing, maka ciphertext beserta tanda tangan digital dan id kendaraan akan dikirim ke database server melalui jaringan internet. Database server pada penelitian ini menggunakan VPS (*Virtual Private Server*) dari hostinger.co.id . Selanjutnya, PC TMC akan melakukan proses

dekripsi dan verifying dengan mengambil ciphertext dari database server dan akan ditampilkan plaintextnya pada PC TMC.

3.5 Uji Coba Serangan

Uji coba serangan yang dilakukan adalah dengan menggunakan serangan pasif. Serangan pasif dapat dilakukan dengan menguping data yang dikirim seperti pada Gambar 3.16. Software yang dapat digunakan adalah wireshark.



Gambar 3.16 Skenario Serangan Pasif

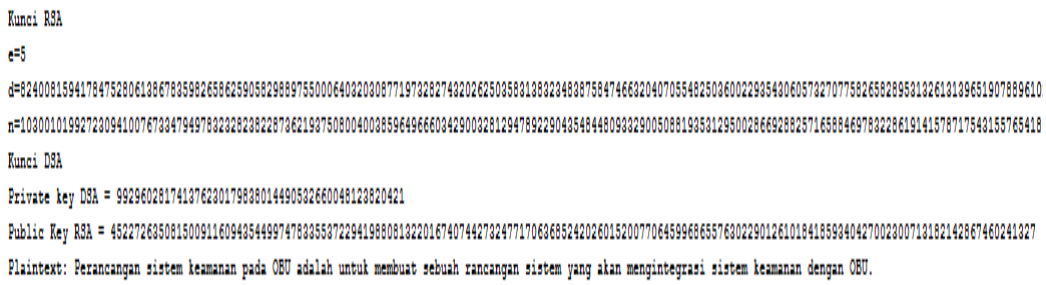
Halaman ini sengaja dikosongkan

BAB 4
HASIL DAN PEMBAHASAN

Pada bagian ini, akan diuraikan mengenai hasil pengujian simulasi dari beberapa skenario pengujian yang dilakukan dan analisis terkait hasil yang didapatkan. Simulasi dilakukan dengan menggunakan *software* Netbeans IDE 8.2. Pengujian yang dilakukan adalah pengujian simulasi sistem keamanan dengan metode gabungan RSA dan *Digital Signature Algorithm,* pengujian simulasi sistem keamanan terintegrasi dengan sistem *Intelligent Transportation System,* dan pengujian sistem keamanan terintegrasi dengan memberi serangan pasif.

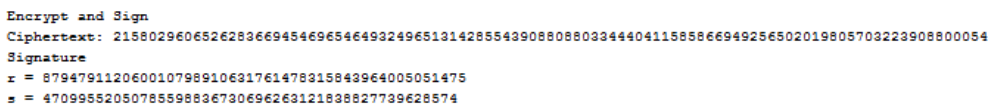
4.1 Hasil Simulasi Sistem Keamanan Menggunakan Metode Gabungan RSA dan *Digital Signature Algorithm*

Pada simulasi sistem keamanan menggunakan metode gabungan RSA dan *Digital Signature* ini, dibuat percobaan dengan menggunakan RSA 1024 dan DSA 512 dengan menggunakan plaintext berupa text “Perancangan sistem keamanan pada OBU adalah untuk membuat sebuah rancangan sistem yang akan mengintegrasi sistem keamanan dengan OBU.” Dengan 134 karakter. Didapatkan hasil untuk pembangkitan kunci seperti pada Gambar 4.1.



Gambar 4.1 Hasil Simulasi Pembangkitan Kunci

Setelah proses pembangkitan kunci, maka proses selanjutnya adalah proses enkripsi dan dekripsi seperti pada Gambar 4.2.



Gambar 4.2 Hasil Simulasi Enkripsi dan Sign

Setelah proses enkripsi dilakukan, proses terakhir adalah dekripsi dan verify seperti pada Gambar 4.3 dimana proses verify membutuhkan tanda tangan digital yang dikirim oleh penerima agar dapat mendekripsi pesan.

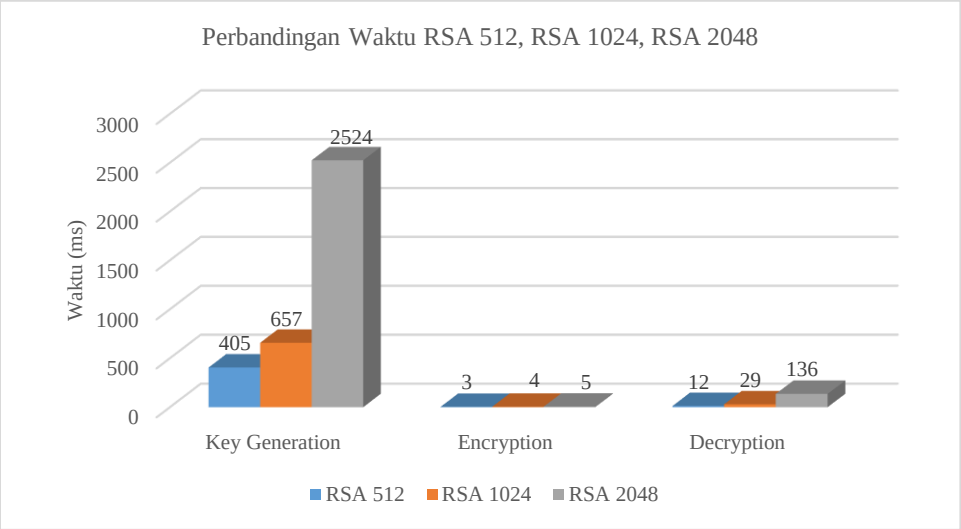
```
Decrypt and Verify
Decrypted String: Perancangan sistem keamanan pada OBU adalah untuk membuat sebuah rancangan sistem yang akan mengintegrasi sistem keamanan dengan OBU.
Valid Signature
```

Gambar 4.3 Hasil Simulasi Dekripsi dan Verify

4.1.1 Hasil dan Analisis Waktu Komputasi

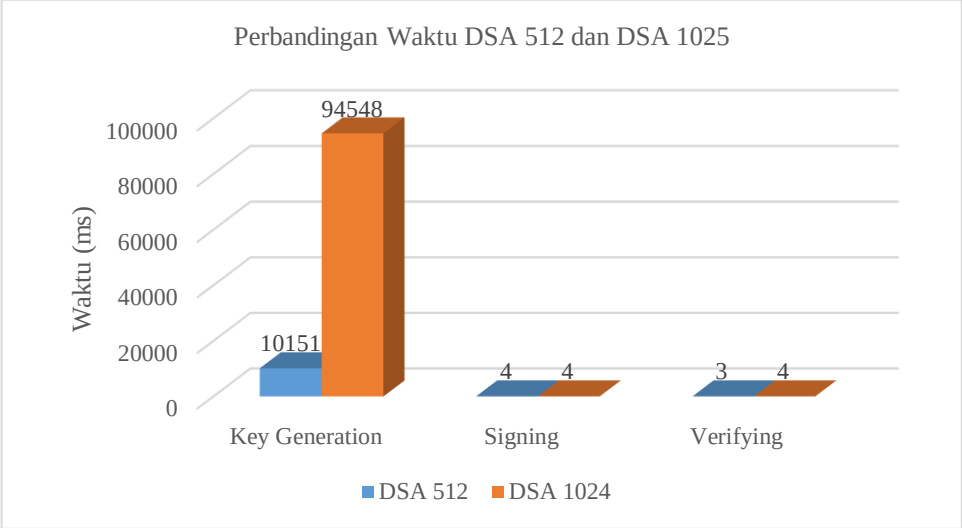
Dari model sistem yang diusulkan, telah dilakukan simulasi untuk mengetahui perbandingan waktu antara RSA 512, RSA 1024, RSA 2048 dan juga DSA 512, DSA 1024 agar dapat memilih dan menyesuaikan berapa bit yang akan digunakan dalam sistem tersebut. Simulasi dilakukan dengan processor Intel(R) Core(TM) i7-3632QM CPU @ 2.20GHz, RAM 4 GB, dan 64-bit Operating System.

Gambar 4.4 menunjukkan perbandingan waktu generate key, enkripsi, dan dekripsi dari RSA dengan berbagai macam bit yaitu 512 bit, 1024 bit, dan 2048 bit. Sehingga dipilih RSA 1024 karena waktu komputasi pada pembangkitan kunci relatif cepat dibandingkan dengan RSA 2048.



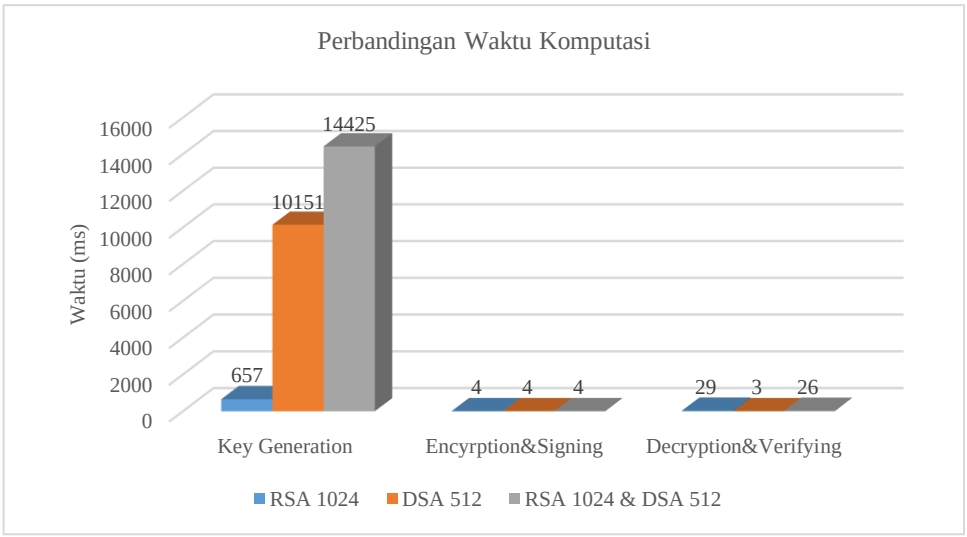
Gambar 4.4 Waktu Komputasi RSA 512, 1024, 2048

Gambar 4.5 adalah perbandingan waktu pembangkitan kunci, signing, dan verifying antara DSA 512 dan DSA 1024. Perbedaan waktu pembangkitan kuncinya sangat jauh. Sedangkan waktu pada proses signing dan verifying hampir sama. Maka dari itu, dipilih DSA 512 karena waktu komputasi pembangkitan kuncinya masih relatif cepat dibandingkan DSA 1024.



Gambar 4.5 Waktu Komputasi DSA 512 dan 1024

Dari hasil yang didapatkan pada simulasi yang telah dilakukan seperti pada Gambar 4.4 dan Gambar 4.5 maka diputuskan untuk menggunakan RSA 1024 dan DSA 512 bit. Berikut ini adalah perbandingan waktu antara pembangkitan kunci, enkripsi dan signing, dekripsi dan verifying antara RSA 1024, DSA 512, dan kombinasi antara RSA 1024 dan DSA 512.



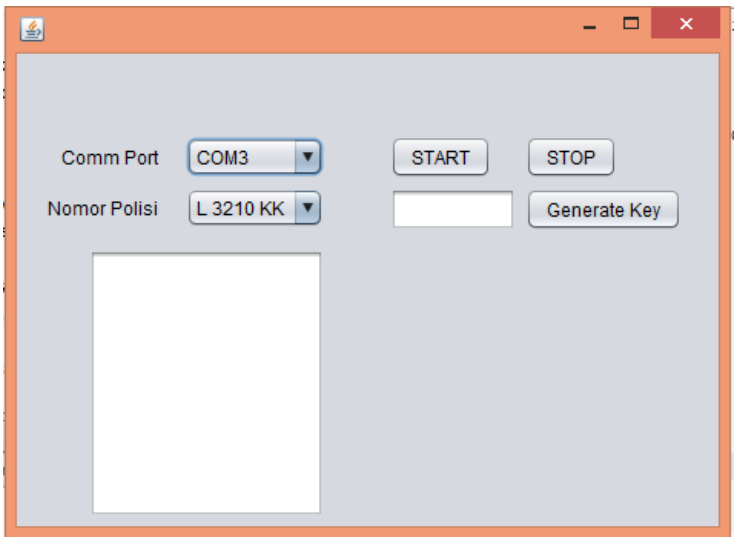
Gambar 4.6 Waktu Komputasi RSA 1024 dan DSA 512

Gambar 4.6 adalah perbandingan dari waktu yang diperlukan untuk pembangkitan kunci, enkripsi dan signing, dekripsi dan verifying. Total waktu yang diperlukan RSA 1024 adalah 690 ms. Total waktu yang diperlukan DSA 512 adalah 10158 ms. Dan total waktu yang diperlukan metode kombinasi RSA 1024 dan DSA 512 adalah 14455 ms.

Dari simulasi yang telah dilakukan, metode kombinasi memiliki waktu pembangkitan kunci yang lebih cepat 33.5% dari waktu pembangkitan kunci RSA dan DSA secara terpisah. Untuk waktu enkripsi dan signing, metode kombinasi memiliki waktu yang sama dengan metode terpisah, dan untuk waktu dekripsi dan verifying memiliki perbandingan waktu lebih lama 18.75%.

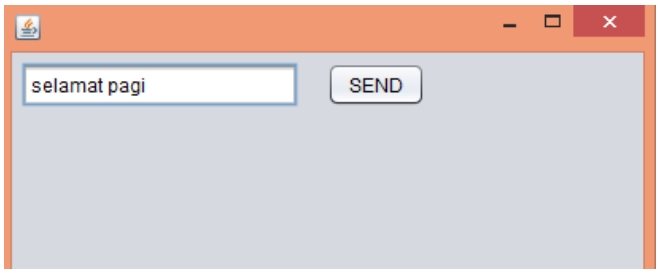
4.2 Hasil Simulasi Sistem Keamanan Terintegrasi dengan *Intelligent Transportation System*

Simulasi sistem keamanan terintegrasi dengan *Intelligent Transportation System* merupakan sistem keseluruhan yang dilakukan penulis. Alur dari sistem seperti yang dijelaskan pada bab 3. Sebelum menjalankan program java, terlebih dahulu pasangkan arduino dengan kabel serial ke PC. Pertama, program main.java dijalankan untuk proses pembangkitan kunci, penerimaan data, dan prose enkripsi dan signing. Gambar 4.7 adalah tampilan awal dari program main.java.



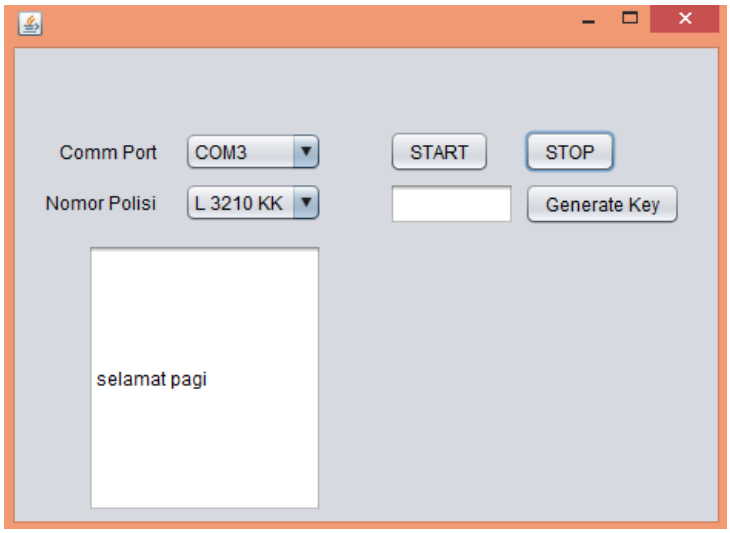
Gambar 4.7 Tampilan Awal Main.java

Setelah tampil seperti pada gambar diatas, langkah pertama yang dilakukan adalah menginputkan nomor polisi pada kolom disebelah tombol generate key dan klik tombol generate key. Setelah itu klik COM3 dan start, maka proses pembangkitan kunci akan dilakukan. Setelah pembangkitan kunci dilakukan, maka tekan tombol start dan program siap menerima data dari arduino dan client udp. Tombol stop adalah untuk menghentikan arduino dalam mengirim data. Dalam simulasi ini, nomor polisi L 3210 KK mendapatkan id kendaraan "1".



Gambar 4.8 Tampilan UDPClient.java

Setelah program utama dijalankan, client dapat mengirimkan pesan yang akan dikirim melalui protokol UDP. Gambar 4.8 merupakan tampilan GUI disisi client untuk mengirimkan informasi berupa text, dan dikirim dengan protokol UDP dengan menekan tombol SEND.



Gambar 4.9 Mengirim Data UDP ke Main.java

Setelah informasi dikirimkan, maka main.java akan menerima informasi tersebut. Dalam hal ini pesan berupa text "selamat pagi" dan akan dienkripsi dengan

data lokasi yang dikirim dengan arduino dengan format informasitext#longitude,latitude;.

```
Plaintext: selamat pagi#-7.3294516,112.7296200;

coba  selamat pagi#-7.3294516,112.7296200;

Panjang String plaintext adalah 38


Encrypt and Sign
Ciphertext: 3171104131869234237773322946007461228599411
Signature
r = 788097367481882231950717613468485658811765261320
s = 1046046408792794357280903107161302399804565469953
```

Gambar 4.10 Hasil Enkripsi dan Signing

Pesan yang telah terenkripsi dikirim bersamaan dengan tanda tangan digital (*r*,*s*) dan juga IDKendaraan ke database server. Gambar 4.11 adalah tabel "kendaraan" pada database server yang berisi id kendaraan, ciphertext, r, s, dan tanggal ciphertext dikirimkan. Dalam simulasi ini id kendaraan yang dipilih adalah "1" dengan nomor polisi L 3210 KK. Dan didapatkan ciphertext yang sama pada Gambar 4.10 dan Gambar 4.11.

IDKendaraan	cyphertext
5	614730979845125228022740735192186105862273107116 70...
5	614730979845125228022740735192186105862273107116 70...
5	614730979845125228022740735192186105862273107116 70...
5	614730979845125228022740735192186105862273107116 70...
5	114167747774884324525043072311991677580956720769 84...
5	614730979845125228022740735192186105862273107116 70...
3	180617505553981683503073610282469789999624661876237...
3	180617505553981683503073610282469789999624661876237...
3	180617505553981683503073610282469789999624661876237...
3	892586461664957720153286896145865005865709229150 18...
3	180617505553981683503073610282469789999624661876237...
3	180617505553981683503073610282469789999624661876237...
3	180617505553981683503073610282469789999624661876237...
6	180617505553981683503073610282469789999624661876237...
5	183806788589370825718077285928142276416106641599 02...
5	687515593696955796144508196469912130507024353357 46...
5	614730979845125228022740735192186105862273107116 70...
5	614730979845125228022740735192186105862273107116 70...
5	614730979845125228022740735192186105862273107116 70...
5	614730979845125228022740735192186105862273107116 70...
1	180617505553981683503073610282469789999624661876237...
1	3171104131869234237773322946007461228599411554986 8...

Gambar 4.11 Chipertext pada Database Server

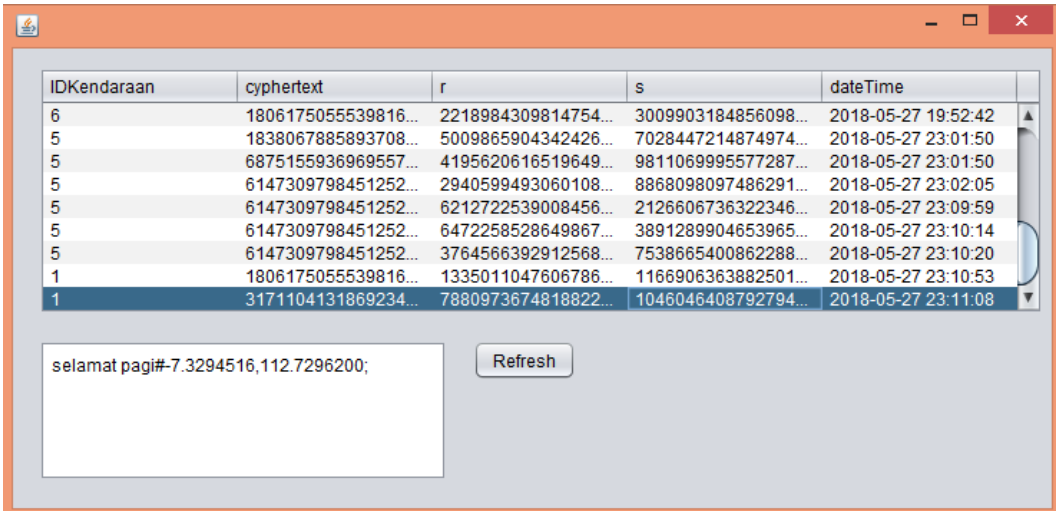
Pesan yang dikirim akan mengalami overhead karena pesan yang dikirim bukan hanya berupa informasi saja tetapi juga terdapat ID kendaraan, ciphertext dari informasi tersebut, tanda tangan digital (*r*,*s*) dan juga *datetime*. Contohnya, seperti pada penjelasan diatas, informasi yang dikirim semula berupa string sepanjang 38 bytes. Setelah dienkripsi maka string tersebut akan menjadi ciphertext dengan panjang 308 bytes. ID kendaraan memiliki panjang 2 bytes, tanda tangan digital (*r*,*s*) memiliki panjang masing-masing 48 bytes, dan *datetime* memiliki panjang 19 bytes. Sehingga jika dijumlahkan secara keseluruhan, maka penambahan overhead yaitu sebesar 425 bytes seperti pada Tabel 4.1 dibawah ini.

Tabel 4.1 Overhead Bit

Pesan yang dikirim	Jumlah bytes
ID kendaraan	2
Ciphertext	308
Signature (*r*,*s*)	96
Datetime	19
Total	425

4.2.1 Hasil dan Analisis Keutuhan Data

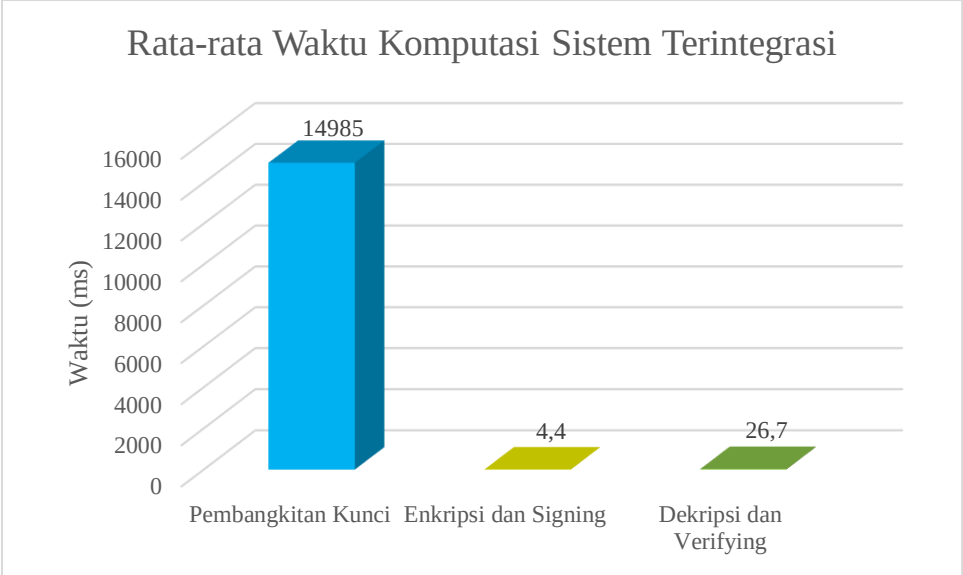
Setelah ciphertext dikirim ke database server, komputer TMC akan mengambil ciphertext tersebut dari database server. Komputer TMC dapat mendekripsi dan memverifikasi ciphertext tersebut jika nilai dari tanda tangan yang diterimanya valid. Gambar 4.12 adalah tampilan GUI dari TMC.java dimana plaintext didapatkan kembali dengan mendekripsi dan memverifikasi ciphertext dan tanda tangan digital. Dapat dilihat bahwa pesan yang diterima sama dengan pesan yang dikirim yaitu selamat pagi#-7.3294516,112.7296200;. Itu berarti data yang diterima memenuhi persyaratan dari layanan keamanan dalam hal keutuhan data.



Gambar 4.12 Tampilan TMC.java

4.2.2 Hasil dan Analisis Waktu Komputasi

Dari hasil simulasi yang telah dilakukan, maka didapatkan rata-rata waktu komputasi dari tiga proses penting pada sistem yang sudah terintegrasi. Rata-rata tersebut diambil dari sepuluh kali percobaan. Tabel akan ditampilkan pada lampiran. Berikut adalah grafik dari rata-rata waktu komputasi yang dibutuhkan pada sistem terintegrasi.



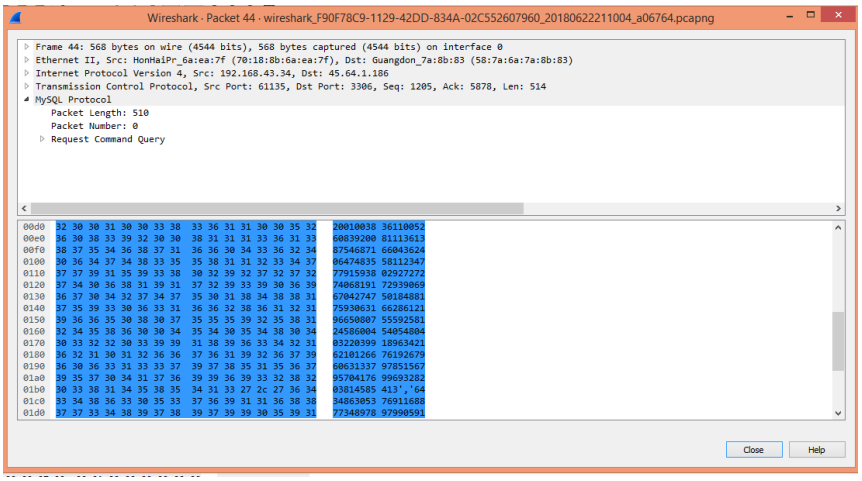
Gambar 4.13 Rata-rata Waktu Komputasi Sistem Terintegrasi

Proses pertama adalah pembangkitan kunci dengan rata-rata waktu 14985 ms, dimana waktu komputasi dalam pembangkitan kunci lebih lama dibandingkan

dengan waktu komputasi yang dibutuhkan dalam pembangkitan kunci sebelum diintegrasikan yaitu 14425 ms. Proses selanjutnya adalah enkripsi dan signing dengan rata-rata waktu 4.4 ms dimana waktu komputasi lebih lama dibandingkan dengan waktu komputasi enkripsi dan signing yang dibutuhkan dalam proses sebelum diintegrasikan yaitu 4 ms. Dan proses terakhir adalah dekripsi dan verifying dengan rata-rata waktu komputasi yang juga lebih lama 0.7 ms dibandingakan dengan waktu komputasi sebelum pengintegrasian sistem. Dari simulasi dan analisa diatas, didapatkan bahwa waktu komputasi dari semua proses akan mengalami penambahan delay waktu dikarenakan proses yang dilakukan juga lebih kompleks.

4.3 Uji Serangan Pasif

Pengujian serangan pasif yang dilakukan adalah sniffing alias menguping. Serangan ini dilakukan oleh *man in the middle* untuk mendapatkan pesan yan dikirim oleh pengirim ke penerima. Dari skenario pada Gambar 3.16, *man in the middle* memanfaatkan perangkat lunak Wireshark untuk melakukan aksinya. Hasil yang didapatkan oleh *man in the middle* tersebut adalah seperti pada Gambar 4.14. Hasil sniffing menggunakan wireshark menunjukkan bahwa data-data yang dikirim masih tetap berupa ciphertext.



Gambar 4.14 Hasil sniffing menggunakan Wireshark

Halaman ini sengaja dikosongkan

BAB 5
KESIMPULAN

5.1 Kesimpulan

Berdasarkan hasil simulasi dan analisis yang dilakukan dapat ditarik beberapa kesimpulan sebagai berikut:

1. Sistem keamanan terintegrasi telah memenuhi persyaratan layanan keamanan dalam hal keutuhan data karena penerima dapat mendekripsi dan memverifikasi ciphertext dengan benar sehingga pesan yang diterima sama dengan pesan yang dikirim.
2. Sistem keamanan yang terintegrasi dengan *Intelligent Transportation System* membutuhkan waktu komputasi lebih lama dari pada sistem keamanan sebelum terintegrasi. Hal ini dikarenakan semakin kompleks program yang dijalankan.
3. Pembangkitan kunci pada sistem terintegrasi memiliki waktu komputasi 3.88% lebih lama (560 ms) dari pembangkitan kunci sebelum terintegrasi.
4. Proses enkripsi dan signing pada sistem terintegrasi memiliki waktu komputasi 10% lebih lama (0.4 ms) dari proses enkripsi dan signing sebelum terintegrasi.
5. Proses dekripsi dan verifying pada sistem terintegrasi memiliki waktu komputasi 2.7% lebih lama (0.7 ms) dari proses dekripsi dan verifying sebelum terintegrasi.

5.2 Saran

1. Hasil akhir dari tesis ini masih berupa simulasi, sehingga kedepannya dapat diimplementasikan secara langsung pada sistem yang ada.
2. Dapat juga dilakukan peningkatan kualitas dan efisiensi dari metoge gabungan RSA-DSA.

Halaman ini sengaja dikosongkan

DAFTAR PUSTAKA

[1] R. Suyuti, "Implementasi Intelligent Transportation System (ITS) untuk Mengatasi Kemacetan Lalu Lintas di DKI Jakarta," *Jurnal Konstruksia,* vol. 3, p. 1, 2012.

[2] N. D. Prasetya, Implementasi Sistem Pengamanan Dokumen Penawaran EProcurement Menggunakan Digital Signature Algorithm (DSA), Surabaya: Politeknik Elektronika Negeri Surabaya, 2014.

[3] U. Somanni, K. Lakhani and Manish Mundra, "Implementing Digital Signature with RSA Encryption Algorithm to Enhance the Data Security of Cloud Computing," in *International Conference on Parallel, Distributed and Grid Computing*, 2010.

[4] R. Sadikin, Kriptografi untuk Keamanan Jaringan, Yogyakarta: Andi Yogyakarta, 2012.

[5] E. S. M. Ranbir Soram, "On the Performance of RSA in Virtual Banking," in *International Symposium on Advanced Computing and Communication (ISACC),* Takyelpat, Imphal, India, 2015.

[6] S. Ezell, "Explaining International IT Application Leadership: Intelligent Transportation System," The Information Technology & Innovation Foundation, Washington, DC, 2010.

[7] R. Munir, "Algoritma RSA dan El-Gamal," Bandung, Institut Teknologi Bandung.

[8] N. Herawati, Perancangan Dan Implementasi DSA (Digital Signature Algorithm) Menggunakan Bahasa Pemrogaman JAVA, Semarang: Universitas Diponegoro, 2006.

[9] A. Solichin, MySQL 5: Dari Pemula Hingga Mahir, Jakarta: Achmatim.net, 2010.

[10] "Netbeans IDE," Oracle Corporation, 2018. [Online]. Available: https://netbeans.org/. [Accessed 24 Mei 2018].

[11] B. M. Wilamowski, Industrial Communication System, London: Taylor and Francis Group, 2011.

[12] A. F. Rochim, "Implementasi Virtual Private Server Menggunakan Xen Hypervisor," *Research Gate,* 2012.

[13] A. Almurayh, “Virtual Private Server,” UCCS, Colorado, Amerika, 2010.

[14] F. Djuandi, Pengenalan Arduino, Tobuku, 2011.

[15] W. Stallings, Network Security Essentials: Applications and Standards Fourt Edition, Prentice Hall, 2011.

LAMPIRAN A
PERHITUNGAN ALGORITMA RSA

1. Pembangkitan Kunci

Misalkan $p = 47$ dan $q = 71$ (keduanya prima). Selanjutnya, hitung nilai

$$r = p \cdot q = 3337$$

dan

$$\phi(r) = (p - 1)(q - 1) = 3220.$$

Pilih kunci publik $SK = 79$, karena 79 relatif prima dengan 3220. *PK* dan *r* dapat dipublikasikan ke umum.

Selanjutnya akan dihitung kunci dekripsi *SK* seperti yang dituliskan pada langkah instruksi 5 dengan menggunakan persamaan 2.15,

$$SK = \frac{1 + (m \times 3220)}{79}$$

Dengan mencoba nilai-nilai $m = 1, 2, 3, \ldots$, diperoleh nilai *SK* yang bulat adalah 1019. Ini adalah kunci dekripsi yang harus dirahasiakan.

2. Enkripsi

Misalkan plainteks yang akan dienkripsikan adalah

X = HARI INI

atau dalam sistem desimal (pengkodean ASCII) adalah :

7265827332737873

Pecah *X* menjadi blok yang lebih kecil, misalnya *X* dipecah menjadi enam blok yang berukuran 3 digit:

$x_1 = 726$ $x_4 = 273$

$x_2 = 582$ $x_5 = 787$

$x_3 = 733$ $x_6 = 003$

Nilai-nilai x_i ini masih terletak di dalam rentang 0 sampai 3337 – 1 (agar transformasi menjadi satu-ke-satu). Blok-blok plainteks dienkripsikan sebagai berikut:

$$726^{79} \bmod 3337 = 215 = y_1$$
$$582^{79} \bmod 3337 = 776 = y_2$$
$$733^{79} \bmod 3337 = 1743 = y_3$$
$$273^{79} \bmod 3337 = 933 = y_4$$
$$787^{79} \bmod 3337 = 1731 = y_5$$
$$003^{79} \bmod 3337 = 158 = y_6$$

Jadi, cipherteks yang dihasilkan adalah

$$Y = 215\ 776\ 1743\ 933\ 1731\ 158.$$

3. Dekripsi

Dekripsi dilakukan dengan menggunakan kunci rahasia

$$SK = 1019$$

Blok-blok cipherteks didekripsikan sebagai berikut:

$$215^{1019} \bmod 3337 = 726 = x_1$$

$$776^{1019} \bmod 3337 = 582 = x_2$$

$$1743^{1019} \bmod 3337 = 733 = x_3$$

…

Blok plainteks yang lain dikembalikan dengan cara yang serupa. Akhirnya kita memperoleh kembali plainteks semula

$$P = 7265827332737873$$

yang dalam karakter ASCII adalah [7]

$$P = \text{HARI INI.}$$

LAMPIRAN B
Perhitungan Algoritma DSA

1. Pembangkitan Kunci

1. Dibangkitkan p=23 dan q=11, p dan q harus prima dan q < p
   Harus memenuhi syarat (p-1) mod q = 0
2. Nilai h, nilai integer sembarang,secara acak didapatkan h=10
   Harus memenuhi syarat 1 < h < p-1
3. Maka untuk g didapatkan
   $g = h^{(p-1)/q} \bmod p$
   $g = 10^{(23-1)/11} \bmod 23 = 10^2 \bmod 23 = 8$
4. Tentukan kunci privat x
   Harus bulat x < q, tidak harus prima → Rahasia
   Misal x = 6
5. Hitung kunci publik y
   $y = g^x \bmod p = 8^6 \bmod 23 = 13$
6. Didapatkan pasangan kunci:
   Kunci Publik : y = 13
   Kunci Privat : x= 6

2. Proses Pemberian Tanda Tangan

1. Untuk sebuah pesan m, misalnya hasil fungsi hash H(m) = 16
2. Tentukan secara acak k
   Harus memenuhi syarat k < q
   Misal, k = 5
3. Hitung k^{-1} dari invers multiplikatif k
   $k * k^{-1} \equiv 1 \bmod q$
   $5 * k^{-1} \equiv 1 \bmod 11$
   Didapat $k^{-1} = 9$
4. Hitung r
   $r = (g^k \bmod p) \bmod q$
   $r = (8^5 \bmod 23) \bmod 11 = 16 \bmod 11 = 5$

5. Hitung s

   $s = (k^{-1} (H(m) + x * r)) \bmod q$

   $s = (9 * (16 + 6 * 5)) \bmod 11 = 414 \bmod 11 = 7$

   Kirimkan pesan m, r=5, dan s=7 bersama-sama

3. Proses Verifikasi Tanda Tangan

1. Hitung s^{-1}

   $s * s^{-1} \equiv 1 \bmod q$

   $7 * s^{-1} \equiv 1 \bmod 11$, didapat $s^{-1} = 8$
2. Hitung w

   $w = s^{-1} \bmod q$

   $w = 8 \bmod 11 = 8$
3. Hitung u1

   $u1 = (H(m) * w) \bmod q$

   $u1 = (16 * 8) \bmod 11 = 128 \bmod 11 = 7$
4. Hitung u2

   $u2 = (r * w) \bmod q$

   $u2 = (5 * 8) \bmod 11 = 40 \bmod 11 = 7$
5. Hitung v

   $v = ((g^{u1} * y^{u2}) \bmod p) \bmod q$

   $v = ((8^7 * 13^7) \bmod 23) \bmod 11 = 16 \bmod 11 = 5$

Karena v=5, dan r=5 → SAMA, maka pesan m adalah ASLI [7]**.**

LAMPIRAN C
LISTING PROGRAM

3. Program Main.java

Program client berisi tentang pembangkitan kunci, penerimaan data dari arduino, penerimaan data dari protokol UDP, dan proses enkripsi dan signing.

```
package rsaclient;
import data.Packet;
import data.PacketHandler;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.IOException;
import java.math.BigInteger;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.scene.control.Label;
import javax.swing.JOptionPane;
import jssc.SerialPort;
```

```
import static jssc.SerialPort.MASK_RXCHAR;
import jssc.SerialPortEvent;
import jssc.SerialPortEventListener;
import jssc.SerialPortException;
import jssc.SerialPortList;
import server.ThreadedUDPServer;
import sign.generate;
import sign.sign;
import sign.verify;

public class Main extends javax.swing.JFrame {
    SerialPort arduinoPort = null;
    ObservableList<String> portList;
    Label labelValue;
    Boolean isReceiveData = false;
    String sql="", dataSerial="", dataUDP="";
    rsa rsa = new rsa(1024);
    generate generate = new generate();
    sign sign = new sign();
    verify verify = new verify();
    Connection con = null ;
    Statement statement;
    String skstring,pkstring,pstring,qstring,gstring;
    String estring,dstring,nstring,xstring;
    BigInteger e,d,n,sk,pk,p,q,g;
    int IDKendaraan;
    DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy/MM/dd
HH:mm:ss");
public void RunServer() {
        server = new ThreadedUDPServer(1337);
        server.receive(new PacketHandler() {
            @Override
```

```
            public void process(Packet packet) {
                String data = new String(packet.getData()).trim();
                System.out.println(data.trim());
                ThreadedUDPServer.CLIENTS.add(packet.getConnection());
                reply(new Packet("OK".getBytes(), packet.getAddr(),
packet.getPort()));
                txtData.setText(data);
                dataUDP = data;
                isReceiveData = true;
            }
        });
        }
        public void reply(Packet packet) {
            server.broadcast(new String(packet.toString()).getBytes());
        }

    private void KoneksiDB(){
        try {
            Class.forName("com.mysql.jdbc.Driver");
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/dbkunci",
"root", "");
            }
        catch (ClassNotFoundException | SQLException e) {
            JOptionPane.showMessageDialog(null, "ERROR \n Gagal Memuat
Database");
        }
    }

    public Main() {
        initComponents();
        start();
        loadcombo();
```

```
    jComboBox1.addActionListener (new ActionListener () {
     @Override
     public void actionPerformed(ActionEvent e) {
        System.out.println(jComboBox1.getSelectedItem().toString());
     }
    });

    cbNoPol.addActionListener (new ActionListener () {
     @Override
     public void actionPerformed(ActionEvent e) {
        if(cbNoPol.getItemCount()>0){
           System.out.println(cbNoPol.getSelectedItem().toString());
           getIDKendaraan(cbNoPol.getSelectedItem().toString());
        }
     }
});
  }
  void loadcombo() {
     try
     {
        cbNoPol.removeAllItems();
        PreparedStatement pst=null;
        ResultSet rs=null;
        Class.forName("com.mysql.jdbc.Driver");
        con = DriverManager
        .getConnection("jdbc:mysql://localhost/dbkunci?"
         + "user=root&password=");
        String charitysql = "SELECT noPol FROM kendaraan";
        System.out.println(sql);
        pst = con.prepareStatement(charitysql);
        rs = pst.executeQuery();
       while(rs.next()){
```

```
            cbNoPol.addItem(rs.getString(1));
          }con.close();
        }catch(Exception e)
        {
           System.out.println("Error"+e);
        }
    }

void getInitData(String IDKendaraan ){
        try
        {
           PreparedStatement pst=null;
           ResultSet rs=null;
           Class.forName("com.mysql.jdbc.Driver");
           con = DriverManager
                 .getConnection("jdbc:mysql://localhost/dbkunci?"
                       + "user=root&password=");
 String charitysql = "SELECT IDKendaraan,p,q,g,x,y,e,d,n FROM generatekey
where IDKendaraan LIKE '"+ IDKendaraan+"'";
           System.out.println(charitysql);
           pst = con.prepareStatement(charitysql);
           rs = pst.executeQuery();
           if(rs.next())
           {
              pstring = rs.getString(2);
              qstring = rs.getString(3);
              gstring = rs.getString(4);
              skstring = rs.getString(5);
              pkstring = rs.getString(6);
              estring = rs.getString(7);
              dstring = rs.getString(8);
              nstring = rs.getString(9);
```

```
            generate.setP(new BigInteger(pstring,26));
            generate.setQ(new BigInteger(qstring,26));
            generate.setG(new BigInteger(gstring,26));
            generate.setX(new BigInteger(skstring,26));
            rsa.setE(new BigInteger(estring,26));
            rsa.setN(new BigInteger(nstring,26));
            System.out.println(IDKendaraan + ","+ pstring + ","+ qstring + ","+
gstring + ","+ skstring + ","+ pkstring  + ","+ estring + ","+ dstring + ","+ nstring );
         }con.close();
      }catch(Exception e)
      {
         System.out.println("Error"+e);
      }
   }

 void getIDKendaraan(String nopol){
      try
      {
         PreparedStatement pst=null;
         ResultSet rs=null;
         Class.forName("com.mysql.jdbc.Driver");
         con = DriverManager
               .getConnection("jdbc:mysql://localhost/dbkunci?"
String charitysql = "SELECT IDKendaraan FROM kendaraan where noPol LIKE
'"+ nopol+"'";
         System.out.println(charitysql);
         pst = con.prepareStatement(charitysql);
         rs = pst.executeQuery();
         if(rs.next())
         {
            IDKendaraan = Integer.parseInt(rs.getString(1));
```

```
            System.out.println(IDKendaraan);
        }con.close();
    }catch(Exception e)
    {
        System.out.println("Error"+e);
    }
}
private void detectPort(){
    portList = FXCollections.observableArrayList();
    jComboBox1.removeAllItems();
    String[] serialPortNames = SerialPortList.getPortNames();
    for(String name: serialPortNames){
        System.out.println(name);
        jComboBox1.addItem(name);
    }
}
private void start() {
        detectPort();
}

public boolean connectArduino(String port){
    System.out.println("connectArduino " + port);
    boolean success = false;
    SerialPort serialPort = new SerialPort(port);
    try {
        serialPort.openPort();
        serialPort.setParams(
                SerialPort.BAUDRATE_9600,
                SerialPort.DATABITS_8,
                SerialPort.STOPBITS_1,
                SerialPort.PARITY_NONE);
        serialPort.setEventsMask(MASK_RXCHAR);
```

```
            serialPort.addEventListener(new SerialPortEventListener() {
                @Override
                public void serialEvent(SerialPortEvent serialPortEvent) {
                    if(serialPortEvent.isRXCHAR()){
               try {
               String st = serialPort.readString(serialPortEvent.getEventValue());
               System.out.print(st);
               String tail = "\n";
               if (st.contains(tail )){
               dataSerial+=st;
               if(isReceiveData)
               dataSerial = dataUDP + dataSerial ;
                saveData(dataSerial);
                isReceiveData = false;
                dataSerial="";
                    }
                    else{
                    dataSerial+=st;
                        }
} catch (SerialPortException ex) {
  Logger.getLogger(RSAClientSerial.class.getName()).log(Level.SEVERE,   null,
ex);
             }}}});

            arduinoPort = serialPort;
            success = true;
        } catch (SerialPortException ex) {
            Logger.getLogger(RSAClientSerial.class.getName())
                    .log(Level.SEVERE, null, ex);
            System.out.println("SerialPortException: " + ex.toString());
        }
```

```
        return success;
    }

    public void saveData(String myData)
    {
        String coba1 = "";
        System.out.println("Plaintext: " + myData);
        String chaosp = myData;
        int lenp = chaosp.length();
        int lenSplit = 128;
        int numSplitString = lenp / lenSplit;
        int firstIndeks = 0;
        int lastIndeks = 0;

        List<String> listString = new ArrayList<>();
        List<String> listStringDecript = new ArrayList<>();
        List<BigInteger> listChiper = new ArrayList<>();

        for (int a = 0; a < numSplitString; a++) {
            firstIndeks = lastIndeks;
            lastIndeks = (a + 1) * lenSplit;
            coba1 = myData.substring(firstIndeks, lastIndeks);
            listString.add(a, coba1);
        }
        int lenSplitSisa = lenp - lastIndeks;
        coba1 = myData.substring(lastIndeks, lenp);
        listString.add(numSplitString, coba1);
        System.out.println("coba  " + listString.get(numSplitString));
        for (int x = 0; x <= numSplitString; x++) {
        System.out.println("Panjang String plaintext adalah " +
listString.get(x).length());
        BigInteger plaintext = new BigInteger(listString.get(x).getBytes());
```

```
            BigInteger ciphertext = rsa.encrypt(plaintext);
            listChiper.add(x, ciphertext);
        }
        String strResultChiper = "";
        for (int j = 0; j < listChiper.size(); j++) {
            strResultChiper += listChiper.get(j);
        }
        System.out.println("\n");
        long start2 = System.nanoTime();
        System.out.println("Encrypt and Sign");
        BigInteger[] signed = sign.sign(myData, generate.x, generate.p, generate.q,
generate.g);
        System.out.println("Ciphertext: " + strResultChiper);
        System.out.println("Signature");
        System.out.println("r = " + sign.rs[0]);
        System.out.println("s = " + sign.s);
        long end2 = System.nanoTime();
        System.out.println("Waktu keseluruhan yang diperlukan selama proses
enkripsi dan signing  adalah " + (end2 - start2) / 1000000 + " ms");
        myData="";
        LocalDateTime now = LocalDateTime.now();
        try {
            Class.forName("com.mysql.jdbc.Driver");
            con = DriverManager
                    .getConnection("jdbc:mysql://localhost/dbkunci?"
                            + "user=root&password=");
sql="INSERT INTO dataposition ( IDKendaraan, cyphertext,r,s, dateTime)value"
+ "('"+ IDKendaraan + "','"+ strResultChiper + "','"+ sign.rs[0] + "','"+ sign.s + "','"+
dtf.format(now) + "')";
statement=con.createStatement();
            statement.execute(sql);
            System.out.println(sql);
```

```
            System.out.println("Data berhasil disimpan");
        } catch(ClassNotFoundException | SQLException ex){
            System.out.println(ex.getMessage());
            System.out.println("Data gagal");
        }
    }

    public void disconnectArduino(){
        System.out.println("disconnectArduino()");
        if(arduinoPort != null){
            try {
                arduinoPort.removeEventListener();

                if(arduinoPort.isOpened()){
                    arduinoPort.closePort();
                }

            } catch (SerialPortException ex) {
                Logger.getLogger(RSAClientSerial.class.getName())
                        .log(Level.SEVERE, null, ex);
            }
        }
    }

    @SuppressWarnings("unchecked")
    private void initComponents() {
        jLabel1 = new javax.swing.JLabel();
        jComboBox1 = new javax.swing.JComboBox<>();
        cbNoPol = new javax.swing.JComboBox<>();
        jLabel2 = new javax.swing.JLabel();
        txtData = new javax.swing.JTextField();
        txtNoPol = new javax.swing.JTextField();
```

```
        btnGenerate = new javax.swing.JButton();
        Start = new javax.swing.JButton();
        Stop = new javax.swing.JButton();
setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        addWindowListener(new java.awt.event.WindowAdapter() {
            public void windowClosed(java.awt.event.WindowEvent evt) {
                formWindowClosed(evt);
            }
            public void windowClosing(java.awt.event.WindowEvent evt) {
                formWindowClosing(evt);
            }
        });
        jLabel1.setText("Comm Port");
        jComboBox1.setModel(new     javax.swing.DefaultComboBoxModel<>(new
String[] { "Item 1", "Item 2", "Item 3", "Item 4" }));
        jComboBox1.addItemListener(new java.awt.event.ItemListener() {
            public void itemStateChanged(java.awt.event.ItemEvent evt) {
                jComboBox1ItemStateChanged(evt);
            }
        });
        jComboBox1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jComboBox1ActionPerformed(evt);
            }
        });
        cbNoPol.setName("cbNoPol");
        cbNoPol.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                cbNoPolActionPerformed(evt);
            }
        });
```

```
        jLabel2.setText("Nomor Polisi");
        txtData.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtDataActionPerformed(evt);
            }
        });

        btnGenerate.setText("Generate Key");
        btnGenerate.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnGenerateActionPerformed(evt);
            }
        });

        Start.setText("START");
        Start.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                StartActionPerformed(evt);
            }
        });

        Stop.setText("STOP");
        Stop.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                StopActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
```

```
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRA
ILING)
                .addGroup(layout.createSequentialGroup()
                    .addContainerGap()
                    .addComponent(txtData,
javax.swing.GroupLayout.PREFERRED_SIZE,                    153,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
layout.createSequentialGroup()
                    .addGap(20, 20, 20)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRA
ILING)
                        .addComponent(jLabel1)
                        .addComponent(jLabel2))
                    .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEA
DING, false)
                        .addComponent(cbNoPol,                    0,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jComboBox1, 0, 78, Short.MAX_VALUE))))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 55,
Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEA
DING)
```

```
                .addComponent(txtNoPol,
javax.swing.GroupLayout.PREFERRED_SIZE,                    82,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(Start))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEA
DING)
                .addComponent(Stop)
                .addComponent(btnGenerate))
            .addGap(24, 24, 24))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(53, 53, 53)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BAS
ELINE)
                .addComponent(jLabel1)
                .addComponent(jComboBox1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(Start)
                .addComponent(Stop))
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEA
DING)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BAS
ELINE)
                .addComponent(txtNoPol,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(btnGenerate))
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BAS
ELINE)
                .addComponent(cbNoPol,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(jLabel2)))
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(txtData,    javax.swing.GroupLayout.DEFAULT_SIZE,
174, Short.MAX_VALUE)
            .addContainerGap())
        );
        pack();
    }
    private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {
    }
    private void formWindowClosed(java.awt.event.WindowEvent evt) {
        disconnectArduino();
    }
    private void jComboBox1ItemStateChanged(java.awt.event.ItemEvent evt) {
    }
    private void formWindowClosing(java.awt.event.WindowEvent evt) {
        disconnectArduino(); }
```

```
    private void cbNoPolActionPerformed(java.awt.event.ActionEvent evt) {
    }
    private void txtDataActionPerformed(java.awt.event.ActionEvent evt) {
    }
    private void btnGenerateActionPerformed(java.awt.event.ActionEvent evt) {
    try {
            Class.forName("com.mysql.jdbc.Driver");
            con = DriverManager
                  .getConnection("jdbc:mysql://localhost/dbkunci?"
                        + "user=root&password=");
     sql="INSERT INTO kendaraan ( noPol)value" + "('"+ txtNoPol.getText() + "')";
            System.out.println(sql);
            statement=con.createStatement();
            statement.execute(sql);
            con.close();
            getIDKendaraan(txtNoPol.getText());
            long start1 = System.nanoTime();
            rsa.generateKeys();
            System.out.println("Kunci RSA");
            System.out.println("e=" + rsa.getE());
            System.out.println("d=" + rsa.getD());
            System.out.println("n=" + rsa.getN());
            generate.gen();
            System.out.println("Kunci DSA");
            System.out.println("Private key DSA = " + generate.x);
            System.out.println("Public Key RSA = " + generate.y);
            long end1 = System.nanoTime();
            System.out.println("Waktu keseluruhan yang diperlukan selama proses
generate key  adalah " + (end1 - start1) / 1000000 + " ms");
            e=rsa.getE();
            d=rsa.getD();
            n=rsa.getN();
```

```
            estring=e.toString(26);
            dstring=d.toString(26);
            nstring=n.toString(26);
            sk=generate.x;
            pk=generate.y;
            p=generate.p;
            q=generate.q;
            g=generate.g;
            skstring=sk.toString(26);
            pkstring=pk.toString(26);
            pstring=p.toString(26);
            qstring=q.toString(26);
            gstring=g.toString(26);

            Class.forName("com.mysql.jdbc.Driver");
            con = DriverManager
                    .getConnection("jdbc:mysql://localhost/dbkunci?"
                            + "user=root&password=");
            sql="INSERT INTO generatekey ( IDKendaraan, p, q, g, x, y,e , d, n)value"
+ "('"+ IDKendaraan + "','"+ pstring + "','"+ qstring +"','"+ gstring +"','"+ skstring
+"','"+ pkstring + "','"+ estring +"','"+ dstring + "','"+ nstring +"')";
            System.out.println(sql);
            statement=con.createStatement();
            statement.execute(sql);
            con.close();
            loadcombo();
        } catch (IOException ex) {
            System.out.println("Ada Error" + ex);
        } catch (ClassNotFoundException ex) {
            Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null, ex);
        } catch (SQLException ex) {
            Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null, ex);
```

```
        }
    }
    private void StartActionPerformed(java.awt.event.ActionEvent evt) {
        RunServer();
        disconnectArduino();
        connectArduino(jComboBox1.getSelectedItem().toString());
        getInitData(Integer.toString(IDKendaraan));
    }
    private void StopActionPerformed(java.awt.event.ActionEvent evt) {
        disconnectArduino();
    }
    public static void main(String args[]) {
     try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
java.util.logging.Logger.getLogger(Main.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Main.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Main.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
```

```
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
java.util.logging.Logger.getLogger(Main.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        }
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Main().setVisible(true);
            }
        });
    }
    private javax.swing.JButton Start;
    private javax.swing.JButton Stop;
    private javax.swing.JButton btnGenerate;
    private javax.swing.JComboBox<String> cbNoPol;
    private javax.swing.JComboBox<String> jComboBox1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JTextField txtData;
    private javax.swing.JTextField txtNoPol;
}
```

4. Program UDPClient.java

Program UDPClient.java adalah program yang berisi tentang pengiriman informasi berupa data text yang akan dikirim menggunakan protokol UDP.

```
package rsaclient;
import client.ThreadedUDPClient;
import data.Packet;
import data.PacketHandler;
public class UDPClient extends javax.swing.JFrame {
        private ThreadedUDPClient client;
```

```
    public void test() {
        client = new ThreadedUDPClient("localhost", 1337);
        client.receive(new PacketHandler() {
            public void process(Packet packet) {
                String data = new String(packet.getData()).trim();
                System.out.println(data.trim());
            }
        });
    }
  public UDPClient() {
    initComponents();
    test();
  }
  private void initComponents() {
    txtField1 = new javax.swing.JTextField();
    jButton1 = new javax.swing.JButton();
setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    jButton1.setText("SEND");
    jButton1.addActionListener(new java.awt.event.ActionListener() {
      public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton1ActionPerformed(evt);
      }
    });
    javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
    getContentPane().setLayout(layout);
    layout.setHorizontalGroup(
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
      .addGroup(layout.createSequentialGroup()
        .addContainerGap()
```

```
            .addComponent(txtField1,
javax.swing.GroupLayout.PREFERRED_SIZE,                    174,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(18, 18, 18)
            .addComponent(jButton1)
            .addContainerGap(137, Short.MAX_VALUE))
      );
      layout.setVerticalGroup(
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
         .addGroup(layout.createSequentialGroup()
            .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BAS
ELINE)
               .addComponent(txtField1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
               .addComponent(jButton1))
            .addContainerGap(266, Short.MAX_VALUE))
      );
      pack();
   }
   private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
client.send(txtField1.getText().getBytes());
   }
   public static void main(String args[]) {
      try {
         for          (javax.swing.UIManager.LookAndFeelInfo          info          :
javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
               javax.swing.UIManager.setLookAndFeel(info.getClassName());
```

```
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
java.util.logging.Logger.getLogger(UDPClient.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(UDPClient.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
java.util.logging.Logger.getLogger(UDPClient.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
java.util.logging.Logger.getLogger(UDPClient.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
    }
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new UDPClient().setVisible(true);
        }
    });
  }
  private javax.swing.JButton jButton1;
  private javax.swing.JTextField txtField1;
}
```

5. Program TMC.java

Program TMC.java adalah program yang berisi tentang dekripsi dan verify dari ciphertext yang dikirimkan.

```
package rsaclient;
```

```
import data.Connection;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.math.BigInteger;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.Statement;
import java.util.Vector;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
import sign.verify;
public class TMC extends javax.swing.JFrame {
    int IDKendaraan;
    BigInteger e,d,n,sk,pk,p,q,g;
    BigInteger ciphertext,r,s;
    java.sql.Connection con = null ;
    rsa rsa = new rsa(1024);
    verify verify = new verify();
    Statement statement;
    String[] columnNames = {"IDKendaraan", "cyphertext", "r", "s", "dateTime"};
    String skstring,pkstring,pstring,qstring,gstring;
    String estring,dstring,nstring,xstring;
    String strIDKendaraan, strcyphertext, strr, strs, strdateTime;
    public TMC() {
        initComponents();
        loadData();
        tblData.addMouseListener(new MouseAdapter() {
```

```
        @Override
        public void mouseClicked(MouseEvent e) {
           JTable target = (JTable)e.getSource();
           int row = target.getSelectedRow();
           int column = target.getSelectedColumn();
           // do some action if appropriate column{
           long start1 = System.nanoTime();
          IDKendaraan = Integer.parseInt(tblData.getValueAt(row, 0).toString());
           ciphertext = new BigInteger(tblData.getValueAt(row, 1).toString());
           r = new BigInteger(tblData.getValueAt(row, 2).toString());
           s = new BigInteger(tblData.getValueAt(row, 3).toString());
           ciphertext = new BigInteger(tblData.getValueAt(row, 1).toString());
           ciphertext = new BigInteger(tblData.getValueAt(row, 1).toString());
           getInitData(Integer.toString(IDKendaraan));
           BigInteger plaintext1 = rsa.decrypt(ciphertext);
           String strResult = new String(plaintext1.toByteArray());
           System.out.println("Plaintext1: " + strResult);
            String verified;
           verified = verify.verify(strResult, p, q, new BigInteger(strr), new
BigInteger(strs), g, pk);
           System.out.println(verified);
           long end1 = System.nanoTime();
           System.out.println("Waktu keseluruhan yang diperlukan selama proses
generate key  adalah " + (end1 - start1) / 1000000 + " ms");
           System.out.println("\n");
           jTextArea1.setText(strResult);
System.out.println("row : " + row + " column : "+ column + "data : "+
Integer.parseInt(tblData.getValueAt(row, 0).toString()));
         }
        });
   }
   void getInitData(String IDKendaraan ){
```

```
try
{
   PreparedStatement pst=null;
   ResultSet rs=null;
   Class.forName("com.mysql.jdbc.Driver");
   con = DriverManager
         .getConnection("jdbc:mysql://localhost/dbkunci?"
               + "user=root&password=");
   String charitysql = "SELECT IDKendaraan,p,q,g,x,y,e,d,n FROM
generatekey where IDKendaraan LIKE "'+ IDKendaraan+"'";
   System.out.println(charitysql);
   pst = con.prepareStatement(charitysql);
   rs = pst.executeQuery();
   if(rs.next())
   {
      pstring = rs.getString(2);
      qstring = rs.getString(3);
      gstring = rs.getString(4);
      skstring = rs.getString(5);
      pkstring = rs.getString(6);
      estring = rs.getString(7);
      dstring = rs.getString(8);
      nstring = rs.getString(9);
      p = new BigInteger(pstring,26);
      q = new BigInteger(qstring,26);
      g = new BigInteger(gstring,26);
      pk = new BigInteger(pkstring,26);
      rsa.setD(new BigInteger(dstring,26));
      rsa.setN(new BigInteger(nstring,26));
      System.out.println(IDKendaraan + ","+ pstring + ","+ qstring + ","+
gstring + ","+ pkstring  + ","+ dstring + ","+ nstring );
   }
```

```
          con.close();
        }
        catch(Exception e)
        {
           System.out.println("Error"+e);
        }
    }
    private void loadData() {
        try{
           DefaultTableModel model = new DefaultTableModel(new
String[]{"IDKendaraan", "cyphertext", "r", "s", "dateTime"}, 0);
           PreparedStatement pst=null;
           ResultSet rs=null;
           Class.forName("com.mysql.jdbc.Driver");
           // Setup the connection with the DB
           con = DriverManager
                 .getConnection("jdbc:mysql://localhost/dbkunci?"
                       + "user=root&password=");
           String charitysql = "SELECT * FROM dataposition";
           System.out.println(charitysql);
           pst = con.prepareStatement(charitysql);
           rs = pst.executeQuery();
           ResultSetMetaData metaData = rs.getMetaData();
           int i = 0;
           while (rs.next()) {
              strIDKendaraan = rs.getString("IDKendaraan");
              strcyphertext = rs.getString("cyphertext");
              strr = rs.getString("r");
              strs = rs.getString("s");
              strdateTime = rs.getString("dateTime");
              model.addRow(new Object[]{strIDKendaraan, strcyphertext, strr,
strs,strdateTime});
```

```
            i++;
        }
        tblData.setModel(model);
        if (i == 1) {
            System.out.println(i + " Record Found");
        } else {
            System.out.println(i + " Records Found");
        }
    } catch (Exception ex) {
        Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null, ex);
    }
}
private void initComponents() {
    jScrollPane1 = new javax.swing.JScrollPane();
    tblData = new javax.swing.JTable();
    jScrollPane2 = new javax.swing.JScrollPane();
    jTextArea1 = new javax.swing.JTextArea();
    jButton1 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    tblData.setModel(new javax.swing.table.DefaultTableModel(
        new Object [][] {
            {null, null, null, null},
            {null, null, null, null},
            {null, null, null, null},
            {null, null, null, null}
        },
        new String [] {
            "Title 1", "Title 2", "Title 3", "Title 4"
        }
    ));
    jScrollPane1.setViewportView(tblData);
```

```
        jTextArea1.setColumns(20);
        jTextArea1.setRows(5);
        jScrollPane2.setViewportView(jTextArea1);
        jButton1.setText("Refresh");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });
        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEA
DING)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 701, Short.MAX_VALUE)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE, 285,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(jButton1)
                        .addGap(0, 0, Short.MAX_VALUE)))
                .addContainerGap())
        );
        layout.setVerticalGroup(
```

```
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addContainerGap(13, Short.MAX_VALUE)
                .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE,                    172,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEA
DING)
                    .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jButton1))
                .addGap(19, 19, 19))
        );
        pack();
    }
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        loadData();
    }
    public static void main(String args[]) {
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
```

```
            }
        } catch (ClassNotFoundException ex) {
java.util.logging.Logger.getLogger(TMC.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        } catch (InstantiationException ex) {
java.util.logging.Logger.getLogger(TMC.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(TMC.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(TMC.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        }
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new TMC().setVisible(true);
            }
        });
    }
    private javax.swing.JButton jButton1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JTextArea jTextArea1;
    private javax.swing.JTable tblData;
}
```

Halaman ini sengaja dikosongkan

BIOGRAFI PENULIS

Penulis bernama Farah Jihan Aufa putri sulung dari Bapak Fasihul Lisan dan Ibu Nur Kholifah. Lahir di Kota Surabaya pada tanggal 18 September 1994. Penulis menempuh pendidikan formal di Madrasah Ibtidaiyah Mabadie Khoiri Ummah di Surabaya, SMP Khadijah di Surabaya, dan SMA Khadijah di Surabaya. Penulis telah menyelesaikan studi D-4 program studi Teknik Telekomunikasi di Politeknik Elektronika Negeri Surabaya pada tahun 2015. Pada tahun 2016, penulis melanjutkan studi S-2 di Institut Teknologi Sepuluh Nopember (ITS) Surabaya dengan mengambil bidang keahlian Telekomunikasi Multimedia. Penulis telah mengikuti ujian proposal tesis pada tanggal 5 Juli 2017 dengan judul "**SISTEM KEAMANAN DENGAN METODE ENKRIPSI *RSA* DAN *DIGITSL SIGNATURE ALGORITHM* UNTUK KOMUNIKASI BERGERAK PADA APLIKASI *INTELLIGENT TRANSPORTATION SYSTEM***" dan telah mengikut sidang tesis dengan judul yang sama pada tanggal 5 Juli 2018 sebagai salah satu syarat untuk memperoleh gelar Magister Teknik (M.T).

Nama : Farah Jihan Aufa
Tempat, tanggal Lahir : Surabaya, 18-September-1994
Alamat : Keputran 6/25 Surabaya
Telp/HP : 031-5478509 / 087702176058
E-mail : farrajihan@gmail.com

Motto : YOLO (*You Only Life Once)*